
Colección Electrónica
PROGRAMA
TUS PROPIAS
AVENTURAS
EN TU COMPUTADORA
Ediciones
Plesa
Sm
Ediciones
CON PROGRAMAS
PARA SPECTRUM (48 K), TRS-80
APPLE, DRAGON, VIC
2×81, ORIC, BBC,
Y PET

PROGRAMA TUS PROPIAS AVENTURAS EN TU COMPUTADORA

Jenny Tyler y Les Howarth



ZX81 versión de «La Casa Escondida» por Chris Oxlade



Contenido

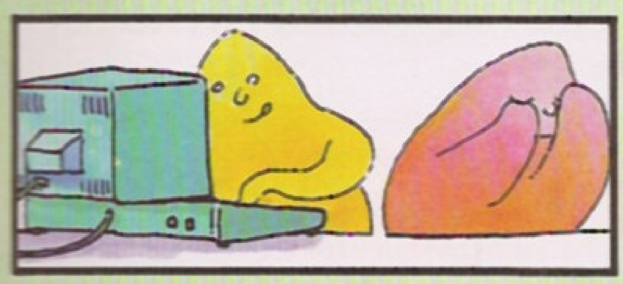
ANOTACION ESPECIAL

Para escribir una aventura has de saber algo sobre el lenguaje BASIC. No obstante, puedes pasarlo bien con este libro si no sabes nada, sólo copiando y jugando a la Casa Encantada, que se escribió para este libro. Si quieres aprender a mejorar tu BASIC hay dos libros de éstos que te servirán: *Programación de Computadoras* y *Guía del BASIC*.

Sobre este libro

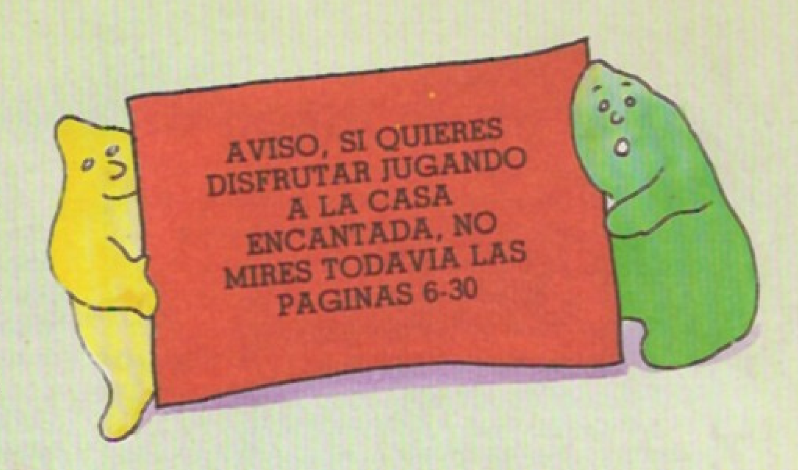
Escribir un programa sobre un juego partiendo de ideas sueltas, puede resultar una labor desalentadora, sobre todo si estás comenzando a programar. Este libro te permite comenzar tan severamente como desees, ofreciéndote un listado de aventuras que puedes copiar y jugar con él, modificarle y añadir todo lo que quieras, o bien usarlo como base para hacer tus propias aventuras.

El juego escrito para este libro se llama *Casa Encantada;* encontrarás el listado principal en las páginas 33-38. Funcionará para cualquier computadora que utilice el estilo Microsoft de BASIC, incluyendo el BBC, el Dragón, el Oric, el TRS-80 y el VIC 20 ampliado, puedes hacerlo cambiar y utilizarlo con el Spectrum 48K (Timex 2000). Una versión especial para el ZX81 (Timex 1000) se ofrece en las páginas 39-45. Dirígete a estas páginas si deseas usar el juego antes de averiguar cómo funciona. Esto puede ser una buena idea si nunca has usado un juego de aventuras.



Si estás acostumbrado a leer listados de programas, probablemente captarás algunas pistas sobre lo que trata el juego mientras lo copias. Para evitar esto, puedes convencer a alguien para que te lo copie.

Ya que el programa es largo y complicado debes tener cuidado al escribirlo. Merece la pena hacerlo lentamente y comprobar cada paso, para no tener que comprobarlos más tarde. Puedes grabar el programa en cinta para usarlo siempre que quieras jugar o para hacer modificaciones.



En las páginas 6-15 averiguarás cómo planificar una aventura y en las páginas 16-27 cómo estructurar y escribir el programa. Puedes encontrar esta parte bastante complicada. No te preocupes por esto, simplemente trabaja cada punto lentamente y con cuidado hasta que hayas comprendido bien las ideas, después pasa al siguiente trozo.

Mientras lees estas páginas puede ser una buena idea escribir un juego como práctica, siguiendo atentamente cada paso. No te preocupes si no es un juego especialmente bueno, te ayudará a comprender cómo funciona el programa y a ver los problemas que tendrás que resolver cuando quieras escribir un juego. Una aventura buena necesita una planificación correcta para que resulte interesante y excitante. Recuerda que no necesitas tocar la computadora hasta que hayas planeado el juego hasta el último detalle.

En la página 46 encontrarás algunos trucos y consejos para escribir aventuras: en la 47 hay respuestas a las preguntas planteadas por todo el libro.

Después de jugar varias veces a la Casa Encantada probablemente querrás hacer cambios. En las páginas 28-31 encontrarás algunas ideas para realizar tus propias versiones.

Las reglas

Si has usado un juego de aventuras anteriormente, ya sabrás lo que puedes esperar de la *Casa Encantada*. Si no, todo lo que debes saber es que la computadora te hará preguntas sobre lo que deseas hacer y debes contestar con dos palabras como mucho.

Es aconsejable recoger cualquier cosa que encuentres por el camino que pueda ser valiosa o útil para resolver los problemas que se te plantearán más tarde. Escribe SCORE para saber qué puntuación tienes y si has ganado.

¿Qué es un juego de aventura?

Un juego de aventura es más o menos una historia en la que el jugador es el héroe. En contra de lo que sucede con los libros donde la sucesión de hechos es fija, en los juegos de aventuras cada vez que juegas sucede algo diferente, ya que eres tú el que vas eligiendo lo que quieres que suceda. El jugador realiza un peligroso viaje a tierras desconocidas según va respondiendo a las cuestiones que le plantea la computadora. El objetivo es sobrevivir a cualquier peligro con el que tengas que enfrentarte y volver con tesoros.

El primer juego de aventura fue escrito en 1976 por William Crowther y Don Woods en una macro-computadora de la Universidad de Stanford, en los EE. UU. Suele denominarse la *Cueva Colosal, Colosal* o simplemente *Aventura,* y actualmente se ofrecen versiones para las computadoras personales. Fue escrito en un idioma científico conocido como Fortran, que no se basa en palabras como el BASIC. Todos los datos para el juego son ordenados y almacenados en disquetes.

Los primeros que jugaron a estas aventuras fueron los profesionales de las computadoras, ya que las computadoras personales no existían. Aún hoy se incluye en los grandes sistemas de computadoras una versión de la aventura de Crowther y Wood para demostrar a la gente que las computadoras pueden ser «amistosas». Estas aventuras basadas en disquetes suelen ocupar más de 250K y resultan complicadas de jugar.

Microaventuras

Desde este comienzo han surgido infinidad de aventuras. Quizá las más populares sean las de Scott Adams, un programador americano que fue el primero en producir una versión de aventura utilizable por los microcomputadores personales. Se llama *Tierra de aventuras,* especial para el TRS-80. Otros juegos de Scott Adams son *Aventura del Pirata, La Cuenta* y *La Pirámide de la Muerte.*

El término «aventura» se utiliza para describir una amplia gama de juegos. El juego de este libro es una aventura tradicional basada en el tipo de aventura de Crowther y Woods. El jugador hace el papel de protagonista en la historia, pero no posee los atributos que suelen dársele en las aventuras clásicas. El jugador utiliza su propia inteligencia, astucia, etcétera, no las que se le suelen asignar a los personales. Al igual que el ajedrez, las aventuras tradicionales son juegos para pensar, no tanto basados en la rapidez de reacciones, sino en la capacidad de resolver problemas.

Aventuras gráficas

La aventura original no utilizaba gráficos, sino que se apoyaba en la imaginación del jugador a la hora de conjurar monstruos y demás seres terroríficos que aparecen durante el juego. Algunas personas piensan que una aventura con gráficos no es una auténtica aventura, aunque actualmente existen algunas aventuras con gráficos muy buenos. Si tienes suficiente memoria puedes añadir gráficos a tus aventuras o a la *Casa Encantada.* Este libro no explica cómo hacer esto, ya que las instrucciones para gráficos varían totalmente de una computadora a otra.



¿Qué tipo de programa es?

Un programa de aventuras es realmente una base de datos. Una base de datos es un sistema de complementación de la computadora que almacena información y permite recurrir a ella de varias formas, posibilitando una amplia gama de usos serios. Un programa de aventuras es una base de datos variables. El jugador lo utiliza variando o renovando información. Según avances con el libro verás cómo algunas palabras pueden utilizarse como «llaves» para poder acceder a ciertos datos. Esta técnica puede usarse para restringir el acceso a determinada información en las bases de datos «serios».

Escribiendo una aventura, puedes aprender algunas técnicas de programación. Lo complicado del programa te hará darte cuenta de lo importante que es planearla detalladamente antes de encender la computadora. También te hará pensar en todas las cosas que una persona utilizando el programa puede intentar hacer. Si encuentras algún método para que la computadora asimile cualquier input (entrada) por tonta que resulte, podrás escribir programas completos que no se interrumpan.

Muchas de las aventuras que puedes comprar en cinta o escritas están elaboradas, aunque sólo sea parcialmente en lenguaje de máquinas. Esto permite introducir más información en la computadora y que el juego se desarrolle más rápidamente. Si conoces algo del lenguaje de máquinas, puedes intentar añadir algunas instrucciones a tus aventuras.

Dónde comenzar

Cuando escribes un juego de aventura estás inventando un mundo de fantasía, donde las reglas las pones tú. Tú decides dónde tiene lugar, qué clase de criaturas viven en él y lo que cada cosa y cada criatura pueden hacer. Ese mundo puede ser una ciudad extraterrestre o un palacio subterráneo con duendes, brujos y gnomos o un castillo misterioso con monstruos. Puede incluso ser un tiempo en el pasado en el que intervienen personajes y hechos de historias reales.

Muchas aventuras utilizan magia de algún tipo. Tú mismo debes decidir hasta qué punto tu aventura se relaciona con la realidad y cuánta magia debe permitirse. Hagas lo que hagas, asegúrate de que las reglas son lógicas, ya que si no, los que la utilicen encontrarán el juego ridículo e incoherente.

Una vez que hayas decidido un tema para tu aventura, necesitarás decidir el objetivo del juego. El jugador puede tener que escapar o volver a algún sitio con tesoros, o puede tener que rescatar a alguien, o encontrar un sitio secreto para hacer algo allí (como desconectar el aparato diabólico del Científico Loco para volar el mundo).

Cálculo de los terrenos

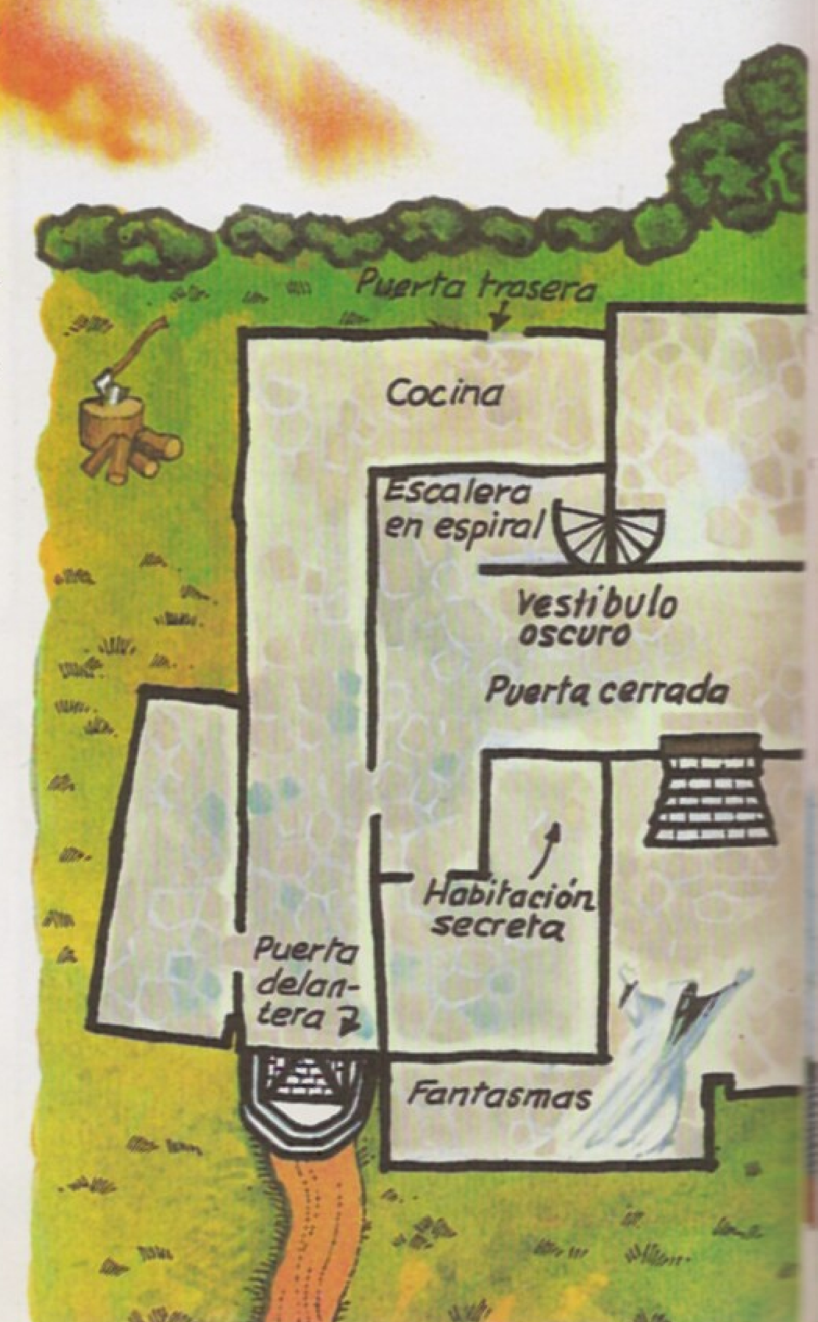
Las zonas o habitaciones que tiene que atravesar el jugador durante la aventura se denominan terrenos. Má adelante verás cómo se numeran para introducirlos en la computadora. De momento, ten en cuenta que el número de terrenos de que puedes disponer depende de la cantidad de memoria que tiene tu computadora. Cuantos más terrenos, más interesante resultará el juego, pero te dejarán menos memoria para describirlos. El juego de este libro tiene 64 terrenos con breves descripciones.

Un terreno puede estar dentro o fuera de las puertas, podría ser una habitación, una caverna, dividir la ruta a lo largo del camino, un área de la selva, en la mitad de un campo, o en cualquier parte que desees. Lo mejor es decidir sobre el número de zonas cercanas y unirlas, pues éstas afectan a la total estructura del juego.

Confección de un mapa

El siguiente paso es hacer un mapa esquemático de tu mundo. No hace falta que sea detallado, pero debe dar una imagen global a escala. Al hacer esto piensa en sitios buenos para esconder tesoros y objetos que los jugadores puedan necesitar.

Aquí vemos un mapa esquemático de la *Casa Encantada* escrita para este libro.





La *Casa Encantada* tiene lugar en un caserón embrujado, situado en el borde de un acantilado. Sus extraños torreones retorcidos destacan frente a un cielo cubierto de niebla. Las paredes no parecen rectas, ni las esquinas cuadradas. No es sorprendente que nadie viva allí —bueno, nadie humano...

Los rumores dicen que el hombre más rico del mundo vivió sus últimos días allí, sorprendentemente nadie vio su cuerpo...

Ideas para temas

Si no se te ocurre una idea para hacer una aventura, piensa en películas o en programas de TV o incluso en libros que hayas leído. Sin embargo, si estás escribiendo un juego para leerlo, recuerda que no debes ajustarte demasiado a la trama, ni utilizar los mismos nombres por razones de derechos de producción.

Estas son algunas ideas para temas.

HISTORIA DETECTIVESCA.—El autor es un detective que investiga un crimen terrible. El objeto del juego es volver a la comisaría con pruebas (las pruebas pueden ser los «tesoros»).

AVENTURA PREHISTORICA.—El jugador viaja a través del tiempo para llegar a los días de las cavernas. El fin es volver al presente con La Piedra, un objeto de gran poder. El jugador tiene que fabricarse sus propias armas, tal y como hacían los hombres de la cavernas. Las bestias prehistóricas y cuevas mágicas pueden ser algunos de los obstáculos.

TEMPLO DEL TERROR.—Las ruinas de un templo construido por una raza misteriosa con fama de conocer el secreto de la vida eterna. Sólo escuchando las cosas que han sucedido allí se te ponen los pelos de punta. El objetivo es escapar con el secreto.

Elaboración de un casillero

El primer paso para convertir tu aventura en un juego de computadora es transformar tu mapa en un casillero. Necesitarás una casilla para cada terreno, por lo que para 64 terrenos, la *Casa Encantada* necesita un casillero de 8×8.

El casillero será la base principal para planear tu aventura, por lo que debes hacerlo grande y claro. Con el tiempo mostrará todos los terrenos y los accesos a ellos, así como todos los tesoros y los objetos utilizados en el juego.

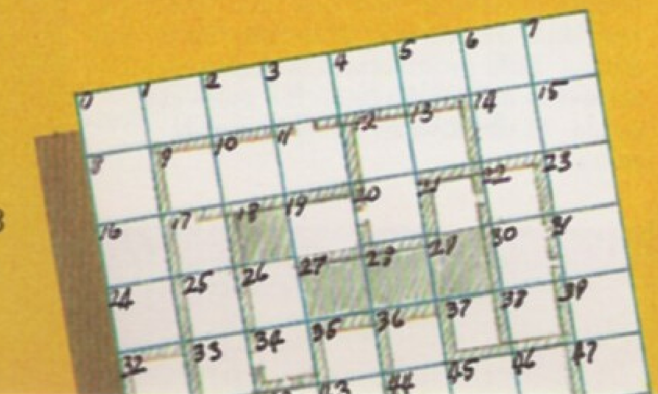
Numera cada casilla comenzando por la esquina superior izquierda. Casi todas las computadoras comienzan a contar en el cero; por tanto, utiliza el cero para nombrar el primer terreno.

0	1	2	3	4	5	6	7
8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23
24	25	26	27	28	29	30	31
32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47
48	49	50	57	52	53	54	55
56	57	58	59	60	61	62	63

Puedes querer cambiar la posición de las paredes y puertas cuando cualcules las rutas que pueden seguir los jugadores, por lo que al principio traza con un lápiz tu mapa sobre el casillero.

Nombra cada casilla con una breve descripción, por ejemplo: «sótano oscuro» o «habitación polvorienta», y luego piensa en las maneras de entrar y salir. La forma habitual de hacer esto es con flechas orientadas como una brújula, siendo el norte la parte superior, el sur la inferior, el este el lado derecho y el oeste el izquierdo.

Si incluyes escaleras o trampillas en tus descripciones, podrás hacer rutas hacia arriba o hacia abajo. Esto hace que el juego sea más interesante y no necesitas un casillero en tres dimensiones.



Cálculo de los caminos

Señala las salidas de cada terreno sobre el casillero.

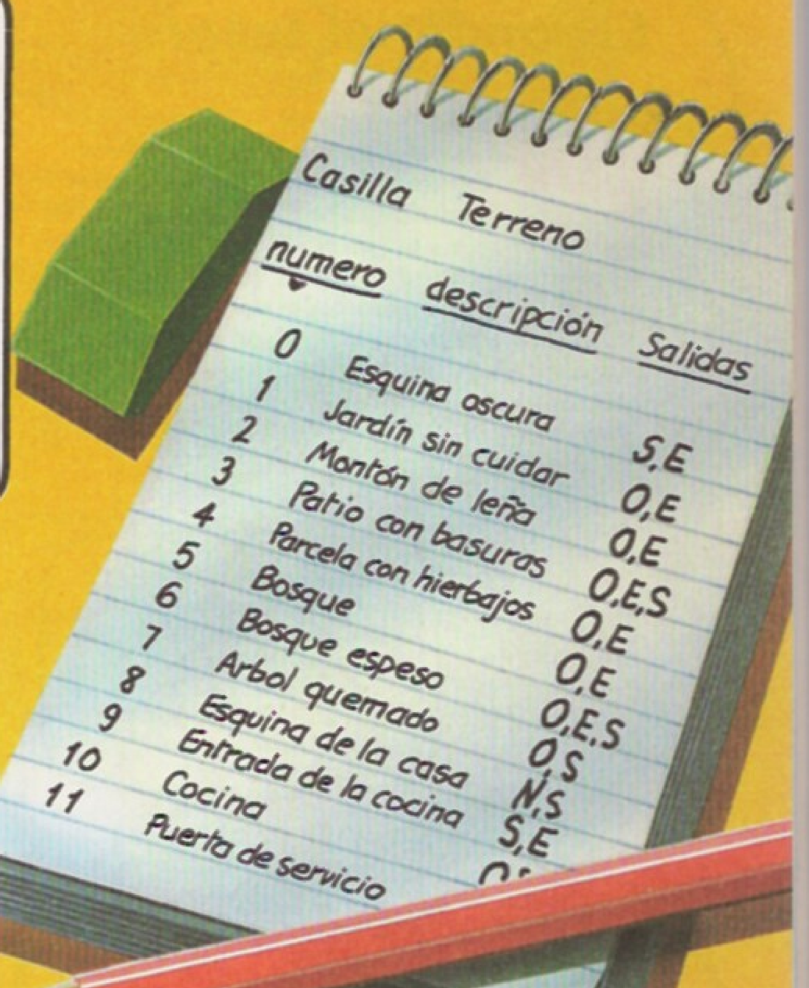
Observa que algunos terrenos de este casillero tienen un solo acceso y una sola manera de salir, por lo que el jugador no puede salir por donde ha entrado. Asegúrate de tener una razón para esto si deseas utilizarlo en tu aventura; estas razones pueden ser mágicas. En la *Casa Encantada* la puerta principal se cierra sola, impidiendo que el jugador salga una vez que ha entrado.

0 ESQUINA OSCURA S▼ E▶	1 JARDIN SIN CUIDAR ◀O E▶	2 MONTON DE LEÑA ◀O E▶	3 E▶ PATIO ◀O S▼
8 ▲N ESQUINA DE LA CASA S▼	9 ENTRADA A LA COCINA S▼ E▶	10 COCINA ◀O E▶	11 ▲N PUERTA DE SERVICIO ◀O
16 ▲N LADO DE LA CASA S▼	17 ▲N PARTE TRASERA VESTIBULO S▼	18 ALCOBA OSCURA S▼ E▶	19 E▶ PEQUEÑA HABITACION OSCURA ◀O
24 ▲N PARED A PUNTO DE DERRUMBAR-SE	25 ▲N PASILLO TENEBROSO S▼	26 ▲N ENTRADA DE LUZ S▼ E▶	27 E▶ VESTIBULO ABOVE-DADO ◀O
32 ARMARIO PARA ABRIGOS S▼	33 ▲N VESTIBULO S▼ E▶	34 ◀O ▲N SALA DE ESPERA S▼	35 HABI-TACION SECRETA PARED FALSA S▼
40 ▲N ROPERO E▶	41 ▲N RECIBIDOR ◀O	42 ▲N BIBLIOTECA E▶	43 ESTUDIO ◀O
48 BARANDA S▼ E▶	49 ▲N PORCHE DELANTERO ◀O S▼	50 TORREON FRONTAL E▶	51 E▶ PASILLO CON PENDIENTE ◀O
56 ▲N VERJA SEMI-DERRUIDA E▶	57 ▲N CAMINO ◀O E▶	58 CAMINO JUNTO A LA VERJA ◀O E▶	59 E▶ BAJO LA TORRE ◀O

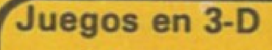
Las marismas de las casillas 53 y 54 son también unidireccionales, ya que la barca se queda atascada. ¿Cuántos caminos de una sola dirección se te ocurren?

Cuando tengas pensados los caminos, marca las paredes y las escaleras y comprueba que tienen sentido con tus caminos. Haz una lista como la que se muestra en la parte derecha, en la que incluirás el número de casilla, su descripción (ésta no tiene que ser necesariamente la versión definitiva) y sus salidas. Mientras escribas el programa esta lista te será muy útil.





Para ayudarte a ver cómo esta información encaja en el programa, puedes completar la lista con el mapa de las páginas 14-15 y comprobarlo con la lista de la página 37.



Juegos en 3-D



Es posible construir auténticos juegos en 3-D con aventuras que suceden en diferentes niveles o pisos de una casa. Para hacer esto necesitas dos o más casilleros de las mismas dimensiones. Tales juegos necesitan mucha memoria (48K es quizá el mínimo válido) y resultan complicados de escribir.

Esconder el tesoro

Una vez trazado el mapa de tu aventura, tendrás que decidir lo que el jugador tiene que hacer. En muchas aventuras el jugador tiene que encontrar objetos valiosos de algún tipo y llevarlos a algún sitio. Estos podrían ser tesoros «reales», como oro y joyas, o podrían ser planos y documentos, o pruebas para resolver algún crimen. Si el propósito de tu juego es rescatar a alguien, considera este alguien como un «tesoro».



También puedes tener obstáculos y trampas en tu juego; no todos han de estar ligados a un tesoro determinado. La *Casa Encantada* tiene, por ejemplo, una marisma.

Añadir «complementos»

Necesitas decidir qué tesoros quieres y dónde esconderlos. Esconder los tesoros posiblemente significará incluir algunos «complementos» en tu plan. Estos pueden ser muebles, alfombras, prendas de vestir, etcétera, que el jugador puede abrir o examinar, pero no llevarse. La *Casa Encantada* tiene un ataúd como «complemento».

Algunas ideas para pensar

¿Puedes pensar en algunos «tesoros» que sean adecuados para estas situaciones?

1. Los cuarteles de un sindicato internacional del crimen.

2. Un planeta lejano más avanzado tecnológicamente que la Tierra.

3. Un laboratorio secreto de investigación.

¿Se te ocurren ahora algunos obstáculos para ellos?

Preparar obstáculos para los jugadores

A continuación debes pensar en los problemas que debe solventar el jugador para encontrar y llevarse los tesoros. Cuanto más ingeniosos y originales sean los problemas que inventes, más interesante resultará el juego. La solución de muchos de los problemas tendrá relación con objetos que el jugador tenga que encontrar para utilizarlos de la forma adecuada. En la página siguiente encontrarás objetos útiles.

Haz una lista de los objetos valiosos y numéralos, esta vez comenzando con 1 (encontrarás el porqué en la página 16). Haz la lista por orden de valor, ya que esto te será útil más adelante para establecer el sistema de puntuaciones. Este es el comienzo de la lista de palabras que deseas que la computadora reconozca.

Haz también una nota de los obstáculos preparados para cada tesoro. Puedes tener un monstruo guardián, por ejemplo, o un duende loco. Los tesoros pueden estar encerrados en cajones o en cajas fuertes. Pueden resultar imposibles de trasladar sin un recipiente que estará escondido en algún lugar. En la parte izquierda hay algunas cuestiones en las que puedes pensar.

Lista de tesoros de la Casa Encantada

numero	Objeto	número de casilla
1	pintura	46
2	anillo	38
3	libro de hechizos	35
4	copa	42
5	pergamino	13
6	monedas	18
7	estatua	28
8	candelabro	42



Anota tus «tesoros» en el plano utilizando los números que les asignaste en la lista. A continuación señala en la lista el número de casilla donde lo has colocado (sólo debes tener un objeto por casilla).

Problema detectivesco

En este juego los objetos valiosos son:

1. Un cabello pelirrojo.
2. Trozo de lana marrón.
3. Huella de pie.
4. Colección de huellas dactilares.
5. Mancha de sangre.
6. Palo pesado de madera.

Los obstáculos con que te encuentras son:

1. El pelo se te perderá si lo llevas tal y como está. Se encuentra en la manga de un abrigo.

2. El trazo de lana no se ve a simple vista, está en el interior de un cajón cerrado.

3. La huella está en una jardinera en la parte externa de una ventana.

4. Las huellas dactilares están sobre una larga mesa junto al cuerpo. No son visibles.

5. La mancha de sangre está sobre la alfombra.

6. Tocar el palo puede significar borrar las huellas dactilares que tenga.

¿Qué soluciones se te ocurren? En la página 47 hay algunas sugerencias, aunque las tuyas pueden ser más interesantes.

Objetos útiles...

Para ayudar al jugador a resolver los problemas que le pones, tendrás que incluir algunas herramientas, armas u otros objetos dentro del juego. El jugador debe encontrar los objetos necesarios y utilizarlos de forma correcta para salvar los obstáculos. Puedes comprobar la inocencia del jugador no incluyendo los objetos más obvios. Por ejemplo, en lugar de una llave, puedes incluir una horquilla para abrir una puerta. Puedes hacer las cosas más difíciles, escondiendo, por ejemplo, una linterna en un lugar y sus pilas en otro, por lo que el jugador tendrá que encontrar ambos lugares.

Añade tus objetos a la lista que comenzaste con tus tesoros. No olvides que algunos de tus tesoros pueden tener utilidad. (No hace falta que los pongas dos veces en la lista.)

Problemas de la *Casa Encantada* y sus soluciones

Estas son las soluciones a algunos de los problemas de la *Casa Encantada* y los objetos que necesitas.

Problema	Solución	Objetos necesarios
Demasiado oscuro para ver	Pon la vela en el candelabro. Enciende la vela con una cerilla	Vela (escondida en un cajón) Cerillas Candelabro (ya está entre los objetos valiosos)
Murciélagos	Fumiga con antimurciélagos	Spray
Habitación secreta	Tira la pared falsa	Hacha
Puerta cerrada	Abrirla	Llave
Fantasmas paralizadores	Absórbelas con la aspiradora	Aspiradora portátil Pilas
Barrera mágica	Utiliza el hechizo mágico	Libro de hechizos (también en la lista de objetos valiosos)
Marisma	Crúzala en barca (sólo se puede usar una vez, porque se atasca en el barro)	Barca
Ventana con rejas	Cava alrededor para sacar los barrotes	Palo
Arcón	Abre la tapa	Nada

Decide dónde deben ir los objetos señalados en tu plano. Resultarán más disimulados si los colocas en sitios donde es normal encontrarlos, por ejemplo un cuchillo en la cocina, un libro en la biblioteca, el hacha junto a la leña. Este es el momento de añadir los complementos descritos en la página anterior. La *Casa Encantada* tiene una mesa en donde se halla la vela. Recuerda que no debes poner los objetos en lugares inaccesibles. No sirve de nada poner la linterna tras una puerta cerrada y la llave de ésta en la oscuridad.

Junto a cada objeto añade el número de casilla. Añade a tu lista cualquier otra palabra (verbos no) que quieres que la computadora entienda. Por ejemplo: norte, sur, fantasmas, murciélagos, arcón. (Acuérdate de añadir los complementos.)

... y lo que se puede hacer con ellos

Ahora que has decidido los objetos que habrá en tu juego, debes decidir lo que los jugadores pueden hacer con ellos. Debes escribir una lista de verbos con los objetos a los que se refieren. Esta debería incluir verbos de movimiento para que los jugadores puedan dar instrucciones de hacia dónde desean ir.

Muchas aventuras están hechas de tal manera que la computadora sólo admite instrucciones de no más de dos palabras de los jugadores. Comprueba la primera palabra con la lista de verbos que has introducido en su memoria y la segunda con el objeto y la dirección que has anotado junto a éste. Gran parte del interés de escribir aventuras es pensar en todas las combinaciones de verbos y objetos que el jugador puede intentar y en decidir la respuesta que la computadora debe dar en cada caso. Los programadores de negocios también deben pensar de esta manera para evitar que sus programas se queden atascados frente a una inesperada respuesta del usuario.

Para tratar verbos (y objetos) que la computadora no localiza en su memoria, puedes incluir respuestas generales como: ¿«Qué haces con él (objeto)?». Agrupa verbos que tengan el mismo significado como coger y tomar. De esta forma ahorrarás memoria al mandar en ambos casos a la computadora a una misma subrutina.

Lista de verbos en la *Casa Encantada*

Numera tus verbos comenzando con 1. (La computadora utilizará el cero para «verbo no encontrado» como verás en la página 16.) AYUDAR y LLEVAR son palabras que se utilizan en cualquier aventura, por lo que debes incluirlas en tu lista de verbos.

También es útil usar abreviaturas para frases como «IR HACIA EL NORTE», etcétera (ver los números 4-9 en la lista). Esto disminuye la cantidad de palabras que el jugador tiene que teclear y hace que el juego resulte más ágil.

N.° de verbo	Verbo	Se refiere a	Acciones (si hay condiciones entre paréntesis)
0	—	—	Para indicar «verbo no encontrado»
1	AYUDA	—	Lista todos los verbos que reconoce la computadora
2	¿LLEVAR?	—	Lista todos los objetos que lleva el jugador
3	IR	DIRECCIONES	Mueve
4	N	—	Abreviatura de «ir HACIA EL NORTE»
5	S	—	Abreviatura de «ir HACIA EL SUR»
6	W	—	Abreviatura de «ir HACIA EL OESTE»
7	E	—	Abreviatura de «ir HACIA EL ESTE»
8	A	—	Abreviatura de «ir HACIA ARRIBA»
9	D	—	Abreviatura de «ir HACIA ABAJO»
10	TOMAR	OBJETOS	Tomar un objeto para llevarlo (el objeto ha de estar en la casilla)
11	TOMAR	OBJETOS	Igual que tomar
12	EXAMINAR	LO QUE SEA	Revela y describe los objetos
13	ABRIR	PUERTA, CAJON	Abre la puerta o el cajón (debe tenerse a llave de la puerta)
14	LEER	LIBROS, HECHIZO	Muestra pistas escritas (debe tenerse el libro de hechizos o estar en la biblioteca)
15	DECIR	CUALQUIER PALABRA	Dice «en alto» las palabras que se escriban. Ej.: un hechizo
16	CAVAR	—	Hace un agujero (debe tenerse la pala y estar en el sótano)
17	GOLPEAR	HACHA	Derrumba la pared falsa (debe tenerse el hacha y estar en el estudio)
18	ESCALAR	CUERDA	Subir o bajar con una cuerda
19	ENCENDER	VELA	Ilumina (debe tenerse la vela, el candelabro y las cerillas)
20	APAGAR	VELA	Apaga la luz (debe hascerse llevando la vela encendida)
21	FUMIGAR	SPRAY	Acaba con los murciélagos del torreón trasero (debe tenerse un spray)
22	USAR	ASPIRADORA	Absorbe los fantasmas (debe tener la aspiradora y las pilas)
23			Abre la puerta (debe tener la llave)
24	DEJAR	CUALQUIER COSA	Abandona objetos (deben tenerse)
25	PUNTUACION	—	Señala la puntuación

Plano principal

El plano principal y las listas que has confeccionado contienen toda la información o datos para tu programa. Este es el plano principal completo para la *Casa Encantada*. (No te preocupes si tu plano no es tan completo como éste.) A continuación averiguarás cómo introducir esta información en la computadora. Pero antes de introducirla asegúrate de que has planeado el juego hasta el último detalle.

0 ESQUINA OSCURA E▶ S▼	1 JARDIN SIN CUIDAR ◀O E▶	2 Hacha (13) ◀O MONTON DE LEÑA E▶
8 N▲ ESQUINA DE LA CASA S▼	9 ENTRADA A LA COCINA E▶ S▼	10 COCINA Cerillas (9) E▶ ◀O
16 N▲ LADO DE LA CASA S▼	17 N▲ TRASTERO DEL VESTIBULO S▼	18 ALCOBA OSCURA Bolsa de monedas (6) E▶ S▼
24 N▲ PARED A PUNTO DE DERRUMBARSE	25 N▲ Aspiradora (10) PASILLO TENEBROSO S▼	26 (11) N▲ ENTRADA DE LUZ E▶ S▼
32 ARMARIO Llave en un bolsillo (18) S▼	33 N▲ VESTIBULO E▶ S▼	34 N▲ ◀O SALA DE ESPERA S▼
40 N▲ ROPERO E▶	41 N▲ ◀O RECIBIDOR Puerta delantera	42 N▲ BIBLIOTECA E▶ Candelabros (8)
48 Escombros BARANDA S▼ E▶	49 N▲ ◀O PORCHE DELANTERO S▼	50 TORREON FRONTAL Copa (14) E▶
56 N▲ VERJA SEMIDESTRUIDA E▶	57 N▲ ◀O PASILLO E▶	58 ◀O CAMINO JUNTO A LA VERJA E▶

Rompecabezas en la aventura

Estas son algunas situaciones con las que pueden encontrarse los jugadores durante el juego. Piensa cuantas soluciones se te ocurren para cada una. En la página 47 encontrarás algunas sugerencias.

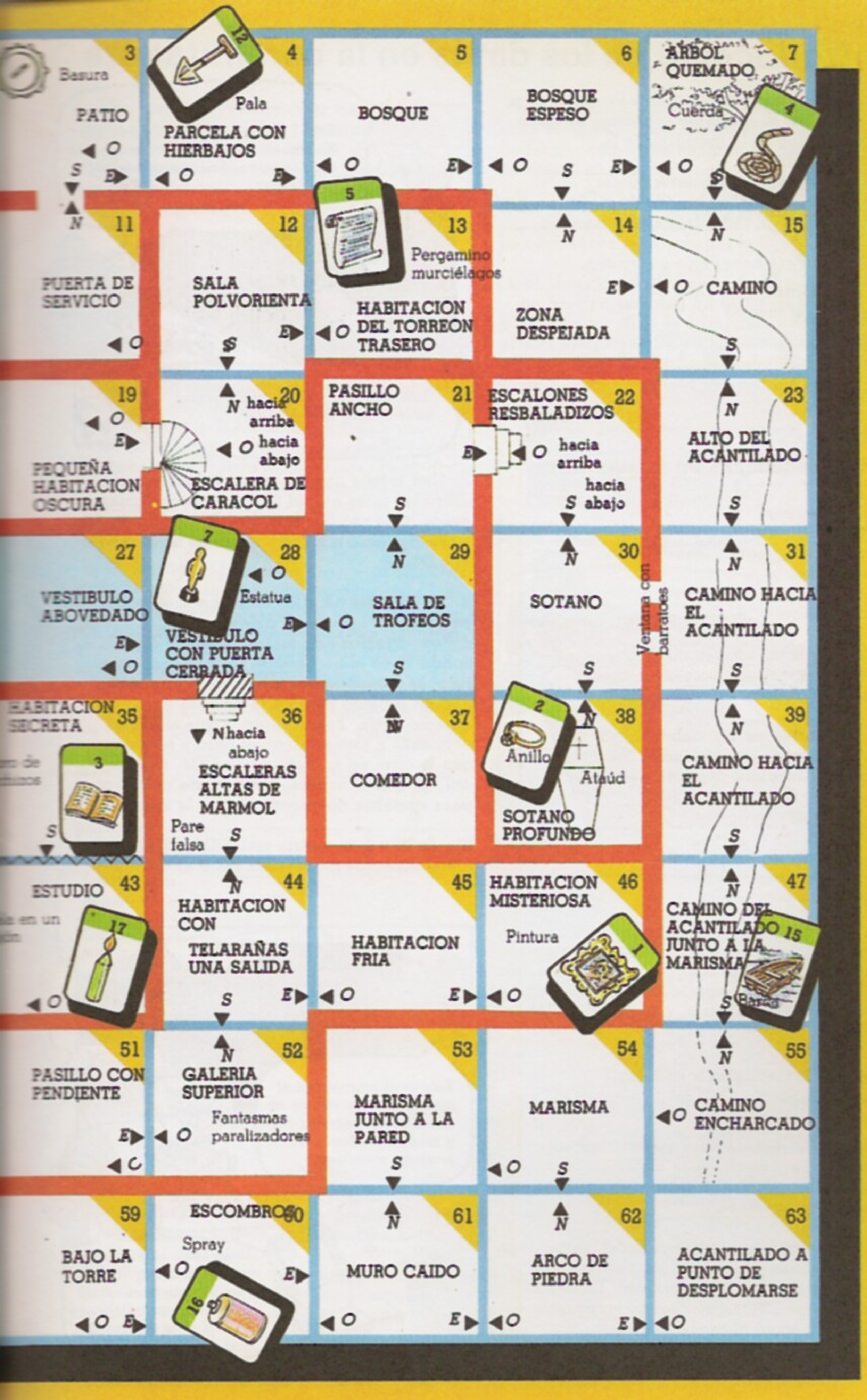
1. Estás atrapado en una habitación de unos tres metros cuadrados. No hay puertas. Hay una gruesa alfombra.

2. Al entrar en una habitación te entra una repentina somnolencia. Vas cargado con una mochila pequeña pero muy pesada y llevas un pañuelo.

3. Te hallas en las almenas de un castillo. Abajo hay un grupo de esclavos furiosos y detrás soldados armados. En la mano tienes un pergamino.

4. Has sido invitado a cenar por un malvado villano. Te desarma totalmente. Se sirven los entremeses y te muestra un control remoto de su arma para destruir el mundo.



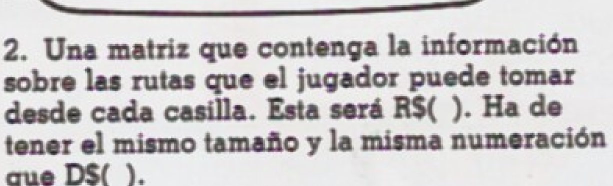


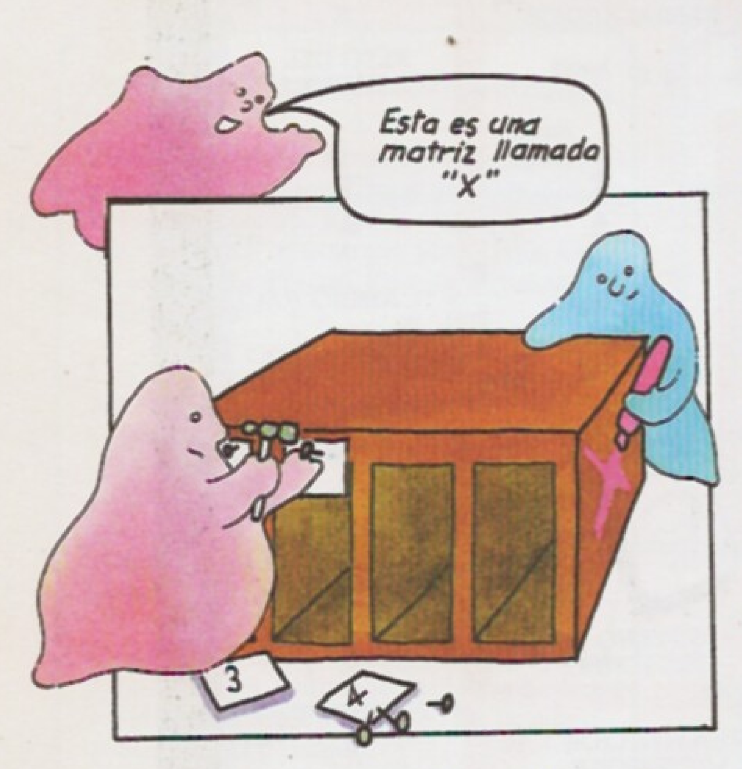
3 Basura PATIO
4 12 Pala PARCELA CON HIERBAJOS
5 BOSQUE
6 BOSQUE ESPESO
7 ARBOL QUEMADO Cuerda 4
11 PUERTA DE SERVICIO
12 SALA POLVORIENTA
13 5 Pergamino murciélagos HABITACION DEL TORREON TRASERO
14 ZONA DESPEJADA
15 CAMINO
19 PEQUEÑA HABITACION OSCURA
20 hacia arriba hacia abajo ESCALERA DE CARACOL
21 PASILLO ANCHO
22 ESCALONES RESBALADIZOS hacia arriba hacia abajo
23 ALTO DEL ACANTILADO
27 VESTIBULO ABOVEDADO
28 7 Estatua VESTIBULO CON PUERTA CERRADA
29 SALA DE TROFEOS
30 SOTANO
Ventana con barrotes
31 CAMINO HACIA EL ACANTILADO
HABITACION SECRETA 35 3
36 hacia abajo ESCALERAS ALTAS DE MARMOL Pared falsa
37 COMEDOR
38 2 Anillo Ataúd SOTANO PROFUNDO
39 CAMINO HACIA EL ACANTILADO
ESTUDIO 43 17
44 HABITACION CON TELARAÑAS UNA SALIDA
45 HABITACION FRIA
46 HABITACION MISTERIOSA Pintura 1
47 CAMINO DEL ACANTILADO JUNTO A LA MARISMA 15 Barca
51 PASILLO CON PENDIENTE
52 GALERIA SUPERIOR Fantasmas paralizadores
53 MARISMA JUNTO A LA PARED
54 MARISMA
55 CAMINO ENCHARCADO
59 BAJO LA TORRE
60 ESCOMBROS Spray 16
61 MURO CAIDO
62 ARCO DE PIEDRA
63 ACANTILADO A PUNTO DE DESPLOMARSE

Introducción de los datos en la computadora

Ahora ya tienes escritos todos los datos para tu aventura. El siguiente problema es cómo introducirlos en la memoria de la computadora.

La computadora necesita que la información sea almacenada de tal forma que pueda acudir a cada dato rápidamente y utilizarlo según el jugador vaya avanzando en el juego. Para hacer esto construyes «matrices» en la memoria de la computadora. Una matriz es como un palomar o como un casillero. A cada matriz se le da un nombre y a cada casilla en ella un número, para que la computadora pueda encontrarla cada vez que te refieras a ella en tu programa.





Antes de introducir información en la computadora debes decidir el tamaño de cada matriz e indicar a la computadora qué nombre y qué reserva ese espacio. Esto se denomina «dimensionar» la matriz, y se escribe DIM en BASIC.

Las matrices para la *Casa Encantada*

La *Casa Encantada* necesita las siguientes matrices para contener la información. Necesitarás matrices similares sea cual sea el tema de tu aventura.

1. Una matriz que contenga las descripciones de las casillas o terrenos. Necesitarás 64 casillas (una para cada terreno). La hemos denominado D$( ) y numerado las casillas del 0 al 63 como en el plano principal.

2. Una matriz que contenga la información sobre las rutas que el jugador puede tomar desde cada casilla. Esta será R$( ). Ha de tener el mismo tamaño y la misma numeración que D$( ).

3. Una matriz para los objetos y demás palabras de la lista de palabras. Si dimensionamos esta matriz DIM O$(W), donde W es el número de las palabras de tu lista, la computadora establecerá una matriz con un espacio para cada palabra y un espacio extra. Esto se debe a que siempre comienza a numerar por el cero y termina con la cifra en la instrucción DIM. Esto es útil, ya que el espacio cero puede utilizarse para «palabra no encontrada en la memoria».

Ej.: Si W=4, la matriz será DIM O$(4) con cinco espacios numerados del 0 al 4.



4. Una matriz de verbos. Esta necesita un espacio para cada verbo y un espacio extra para «verbo no encontrado». Se denomina V$( ) y necesita una dimensión DIM V$(V), donde V es el número de verbos de tu lista.

Más matrices

Los terrenos, los caminos, los objetos y los verbos no son la única información que tiene que almacenarse en la computadora. También necesitas matrices para almacenar información sobre la situación de los objetos, sobre qué objetos lleva el jugador y datos tales como si la luz está encendida o no.

No es necesario volver a almacenar los terrenos y los objetos nuevamente. Para referirse a ellos pueden almacenarse en forma de números para ahorrar espacio. Ej.: el objeto 9 está en el terreno 10.

La matriz L( ) muestra en qué casilla está cada objeto. Sólo necesita espacios para los objetos «accesibles», como la llave, no para los complementos y demás palabras. Si G es el número de objetos accesibles, esta matriz se dimensionará como DIM L(G).

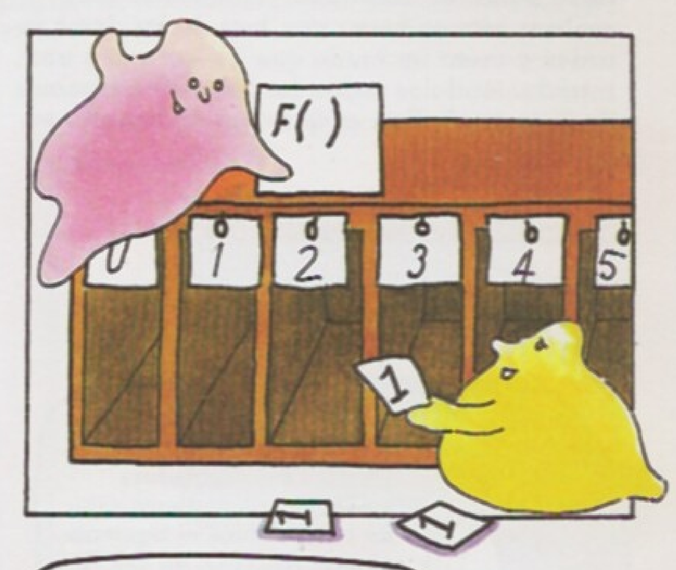
La matriz C( ) es para la información sobre los objetos que lleva el jugador. Esta también requiere espacios sólo para los objetos accesibles, por lo que se dimensionará DIM C(G).



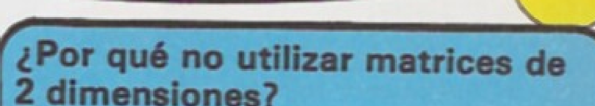
Banderas (Flags)

Además de tener en cuenta los objetos que lleva el jugador, la computadora ha de recordar los cambios que tienen lugar durante el juego; ej.: si la vela está encendida, si la puerta está cerrada o si la llave está visible.

Esto puede hacerse utilizando una matriz F( ), de signos señalados o «banderas» que contenga W espacios; es decir, uno para cada objeto. Colocando 1s y 0s en estos espacios, la computadora puede ver en qué estado se halla el objeto. 0 se utiliza para estado «normal», como puede ser la luz encendida y un objeto no visible.

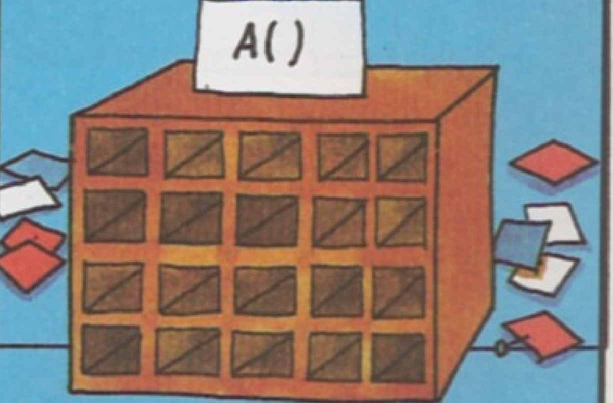


¿Sabías que las computadoras tienen registros de banderas en el CPU que funcionan como esta matriz? Los utilizan para almacenar información sobre lo que está sucediendo mientras se ejecuta un programa.



¿Por qué no utilizar matrices de 2 dimensiones?

Si ya sabías lo que era una matriz, te preguntarás por qué en la *Casa Encantada* se utilizan matrices simples para las descripciones y las salidas en lugar de utilizar matrices en dos dimensiones que serían de esta forma:



La razón es que las matrices de una sola dimensión utilizan menos memoria. Puedes utilizar matrices de dos-dimensiones si lo deseas, en cuyo caso deberías dimensionarlas DIM D$(8,8) y DIM R$(8,8).

Introducción de los datos en las matrices

Habiendo creado zonas de almacenamiento con sus respectivos nombres en la memoria de la computadora, ahora debes decirle qué debe poner en cada una. Un método para realizar esto es hacer una lista de los datos en orden y crear un bucle que los lea uno a uno, introduciéndolos según los lee en los espacios de la matriz*. Esta es la forma de hacerlo en BASIC.

```
DIM O$ (W)
DATA CUADRO, ANILLO, LIBRO DE HECHIZOS, COPA, ETC.
FOR I=1 TO W
READ O$ (I)
NEXT I
```

Primera vez I = 1 por lo que PAINTING se almacena en O$(1). La computadora vuelve por el siguiente valor de I e introduce el siguiente objeto, ANILLO, en O$(2), y así sucesivamente.

Anillo

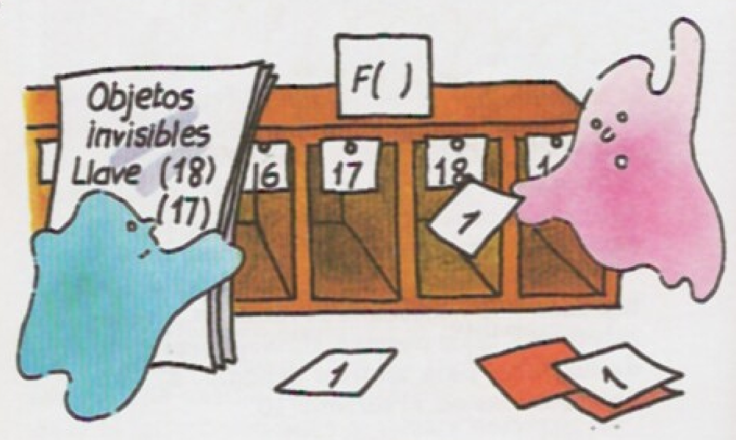
El bucle comienza con el 1, tanto para las matrices numéricas como para la de objetos y la de verbos, por lo que la computadora deja libre el espacio cero. En las matrices de las casillas y de las salidas, el bucle comienza en cero.



Este es el bucle para leer (READ) los datos (DATA) correspondientes a 0$( ). Observa los pasos 1.600 a 2.100 del listado del programa de las páginas 36 y 37 para ver si encuentras los bucles de las demás matrices.

Data para las banderas

El data para la matriz de banderas F( ) consiste sólo de 1s y ceros. Los objetos no visibles al comienzo del juego tienen un uno en su caja en F( ). Cuando los descubre el jugador, la bandera cambia a cero. Todos los demás objetos comienzan con un cero.



Sólo tienes que indicar a la computadora qué cajas de F( ) necesitan un 1. Dejar el resto vacío equivale a rellenarlas con ceros. La manera más fácil de llenar esta matriz es como se muestra en el paso 2090 (página 37).

Puede que hayas notado que algunos espacios de F( ) no se utilizan, ya que algunos objetos no cambian su «estado». Estas banderas de sobra pueden utilizarse para otras cosas. Por ejemplo, en la *Casa Encantada* F(14) (bandera de la cuerda) se utiliza para indicar si el jugador está en lo alto del árbol. La vela necesita dos banderas, una para indicar si está visible y otra para indicar si está encendida. La bandera de osbra F(0) se utiliza para encenderla. Si quieres alguna bandera para algo, utiliza las de palabras que no las necesiten como «norte».

Data para lo que lleva el jugador

Al comenzar el juego el jugador no lleva nada porque la matriz C( ) está vacía. Cuando recoja algo la computadora lo situará en la caja 1. Por tanto, no se necesitará data para la matriz C( ).

Inicialización

Establecer las matrices y llenarlas con datos se denomina «inicialización». En la siguiente sección verás cómo esto encaja en la estructura del programa.

* El ZX81 (Timex 1000) no hace esto, ver página 39.

La estructura del programa

Para lograr una estructura global del programa, primero has de pensar en las funciones que la computadora debe realizar durante el juego.

La estructura del programa suele ser más o menos así. Como ves la parte más amplia son las subrutinas. Ha de existir una subrutina para cada verbo utilizado en el juego. En la página 25 encontrarás más información sobre esto.

Funciones de la computadora

1. Establecer matrices e introducir datos en
ella (INICIALIZACION)
2. Ofrecer la descripción del terreno y las salidas
en el que se haya el jugador (DESCRIPCION)
3. Ofrecer cualquier mensaje o información
adicional como respuesta a las informa-
ciones del jugador (FEEDBACK)
4. Pedir al jugador las siguientes instruccio-
nes (INPUT)
5. Comprobar las instrucciones del jugador con la
lista de verbos y palabras. (ANALISIS DEL INPUT)
6a. Si las instrucciones no tienen sentido, pre-
guntar al jugador y esperar instrucciones
nuevas. (MENSAJES DE ERROR)
6b. Si las instrucciones tienen sentido, ramificar
hacia las subrutinas para desarrollar la
acción adecuada (RAMIFICAR A SUBRUTINAS)
7. Utilizar las subrutinas para llevar a
cabo las instrucciones.
8. Volver a 2.



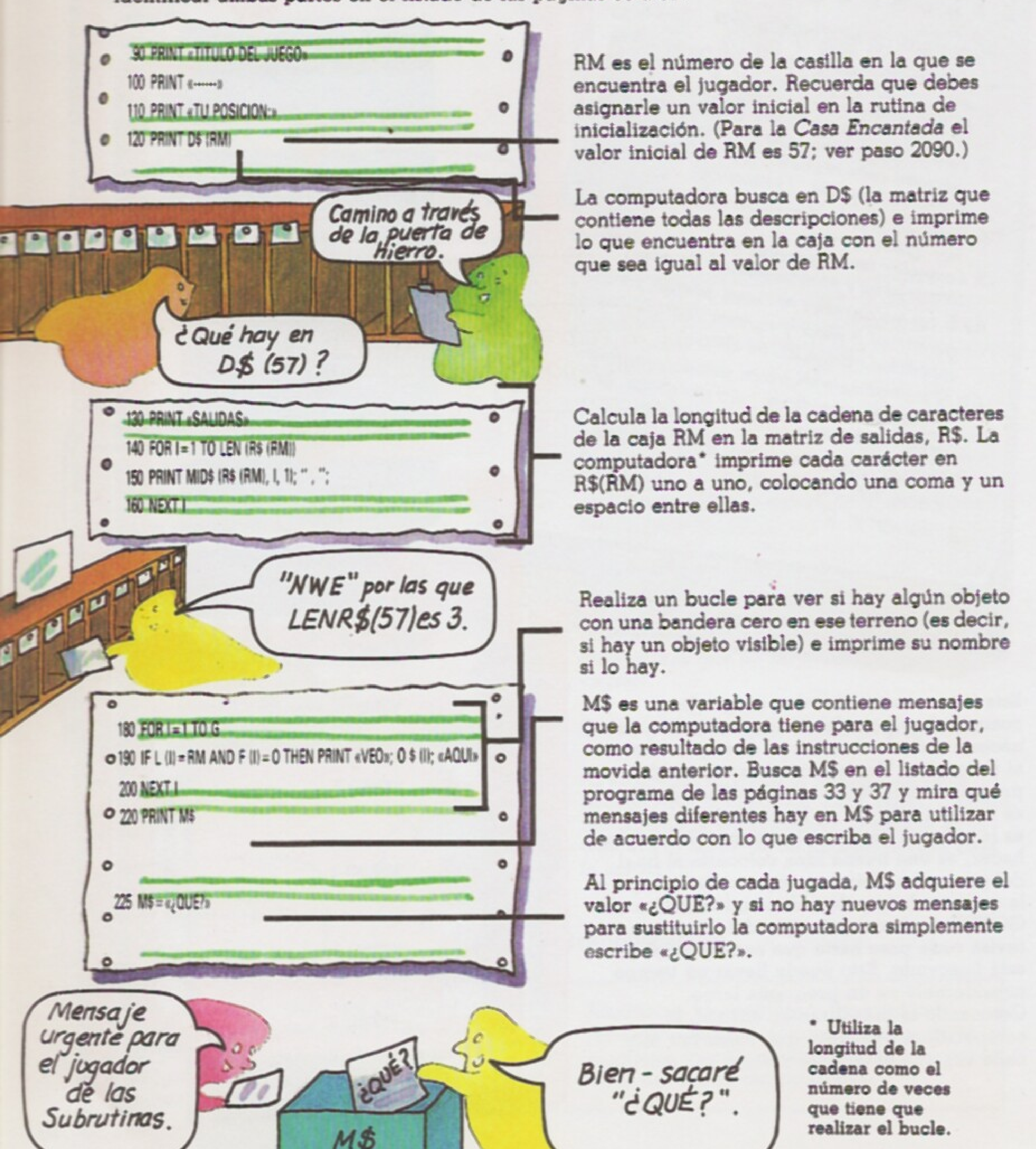
Esta lista muestra el orden en que la computadora hace las cosas, aunque no sea necesariamente el orden en que aparezcan en el programa. Una sección importante del programa es la rutina de inicialización. Esta se usa una sola vez en cada juego, y aunque es la primera cosa que la computadora debe hacer, es una buena idea colocarla al final del programa. Esto es porque cada vez que a la computadora se le ordena un GOTO o un GOSUB vuelve al principio del programa y revisa cada paso hasta que encuentra el que está buscando. Esto puede llevar un tiempo considerable en un programa largo. Colocando la inicialización al final, la computadora no tendrá que pasar por ello cada vez que el jugador realice una movida.

Comienzo del programa

Ahora que tienes una idea general sobre cómo será el programa, puedes comenzar a pensar en cada parte con más detalle. Ya has visto en las páginas 16 y 18 cómo funciona la sección de inicialización. Las ocho siguientes páginas describen como funcionan las demás secciones importantes del programa.

Descripción y feedback

En cada jugada la computadora debe informar al jugador de dónde está y de las direcciones que puede tomar. También debe informarle sobre del resultado de su última instrucción. Estas son las secciones de descripción y feedbäck (búsqueda de información). Mira a ver si puedes identificar ambas partes en el listado de las páginas 35 a 37.



RM es el número de la casilla en la que se encuentra el jugador. Recuerda que debes asignarle un valor inicial en la rutina de inicialización. (Para la *Casa Encantada* el valor inicial de RM es 57; ver paso 2090.)

La computadora busca en D$ (la matriz que contiene todas las descripciones) e imprime lo que encuentra en la caja con el número que sea igual al valor de RM.

Calcula la longitud de la cadena de caracteres de la caja RM en la matriz de salidas, R$. La computadora* imprime cada carácter en R$(RM) uno a uno, colocando una coma y un espacio entre ellas.

Realiza un bucle para ver si hay algún objeto con una bandera cero en ese terreno (es decir, si hay un objeto visible) e imprime su nombre si lo hay.

M$ es una variable que contiene mensajes que la computadora tiene para el jugador, como resultado de las instrucciones de la movida anterior. Busca M$ en el listado del programa de las páginas 33 y 37 y mira qué mensajes diferentes hay en M$ para utilizar de acuerdo con lo que escriba el jugador.

Al principio de cada jugada, M$ adquiere el valor «¿QUE?» y si no hay nuevos mensajes para sustituirlo la computadora simplemente escribe «¿QUE?».

*** Utiliza la longitud de la cadena como el número de veces que tiene que realizar el bucle.**

La sección input

Una característica importante de los juegos de aventuras es como responde la computadora a las instrucciones que escribe el jugador. La *Casa Encantada*, como muchos otros juegos, limita las respuestas del jugador a dos palabras más algunos comandos especiales de una palabra como AYUDA. La siguiente sección de tu programa debe pedir instrucciones al jugador y decir a la computadora lo que debe hacer con ellos.

Para empezar, la computadora ha de dividir la sentencia del jugador en sus dos palabras para poder compararlas con las que tiene en su memoria. La rutina para dividir palabras que se utilizan en la *Casa Encantada* funciona examinando la sentencia del jugador hasta encontrar un espacio libre entre las letras. La encontrarás aquí debajo con algunos pasos extras; puedes copiarla por sí sola y ver cómo funciona*.

```
100 CLS
110 PRINT «POR FAVOR ESCRIBE ALGO»
120 INPUT Q$
130 V$=« »
140 W$=« »
```

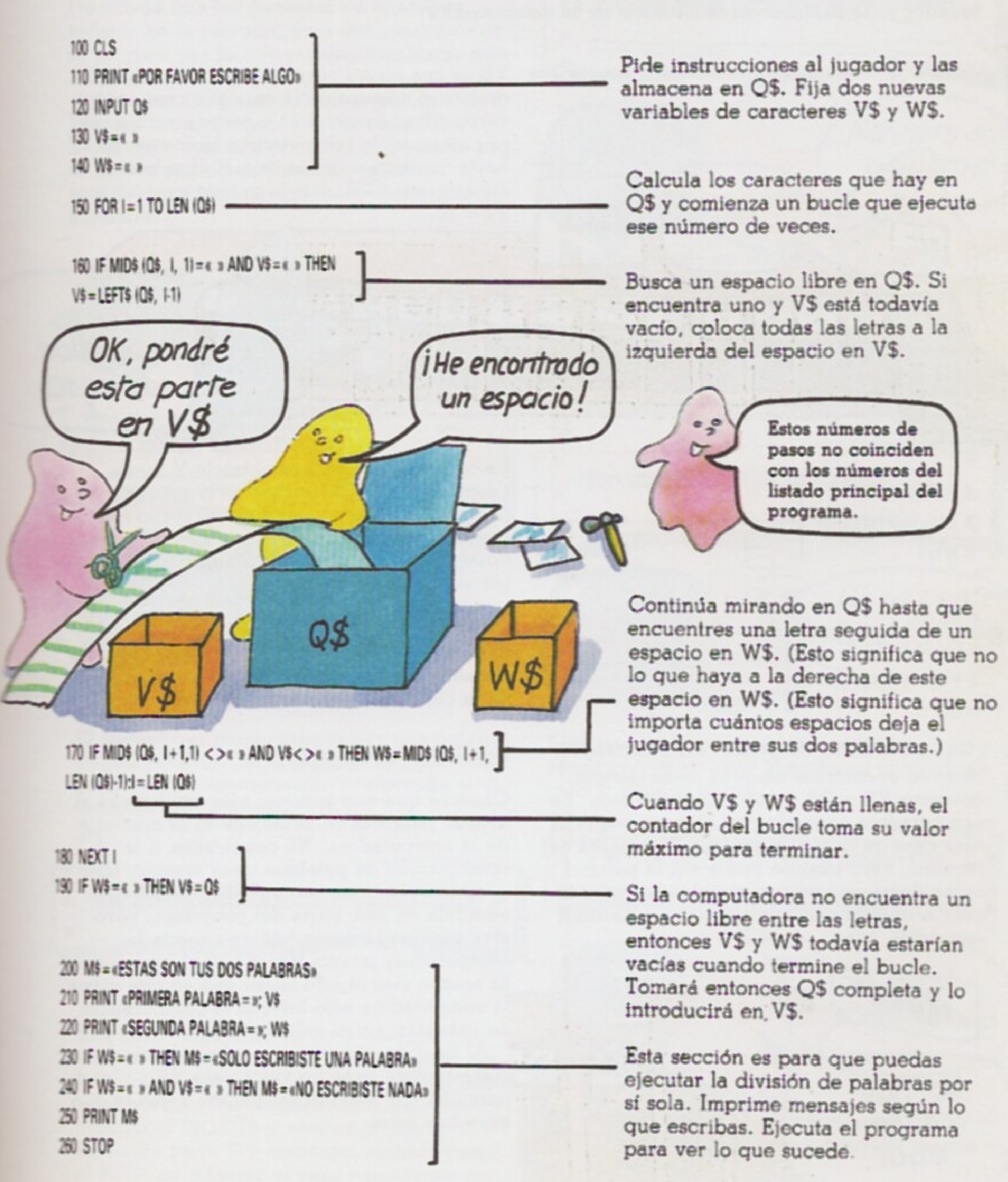
Pide instrucciones al jugador y las almacena en Q$. Fija dos nuevas variables de caracteres V$ y W$.

```
150 FOR I=1 TO LEN (Q$)
```

Calcula los caracteres que hay en Q$ y comienza un bucle que ejecuta ese número de veces.

```
160 IF MID$ (Q$, I, 1)=« » AND V$=« » THEN
V$=LEFT$ (Q$, I-1)
```

Busca un espacio libre en Q$. Si encuentra uno y V$ está todavía vacío, coloca todas las letras a la izquierda del espacio en V$.




```
170 IF MID$ (Q$, I+1,1) <>« » AND V$<>« » THEN W$=MID$ (Q$, I+1,
LEN (Q$)-1):I=LEN (Q$)
```

Continúa mirando en Q$ hasta que encuentres una letra seguida de un espacio en W$. (Esto significa que no lo que haya a la derecha de este espacio en W$. (Esto significa que no importa cuántos espacios deja el jugador entre sus dos palabras.)

Cuando V$ y W$ están llenas, el contador del bucle toma su valor máximo para terminar.

```
180 NEXT I
190 IF W$=« » THEN V$=Q$
```

Si la computadora no encuentra un espacio libre entre las letras, entonces V$ y W$ todavía estarían vacías cuando termine el bucle. Tomará entonces Q$ completa y lo introducirá en V$.

```
200 M$=«ESTAS SON TUS DOS PALABRAS»
210 PRINT «PRIMERA PALABRA=»; V$
220 PRINT «SEGUNDA PALABRA=»; W$
230 IF W$=« » THEN M$=«SOLO ESCRIBISTE UNA PALABRA»
240 IF W$=« » AND V$=« » THEN M$=«NO ESCRIBISTE NADA»
250 PRINT M$
260 STOP
```

Esta sección es para que puedas ejecutar la división de palabras por sí sola. Imprime mensajes según lo que escribas. Ejecuta el programa para ver lo que sucede.

* **Esto no funcionará en computadoras Sinclair (Timex). Ver páginas 38 ó 39.**

Análisis del input

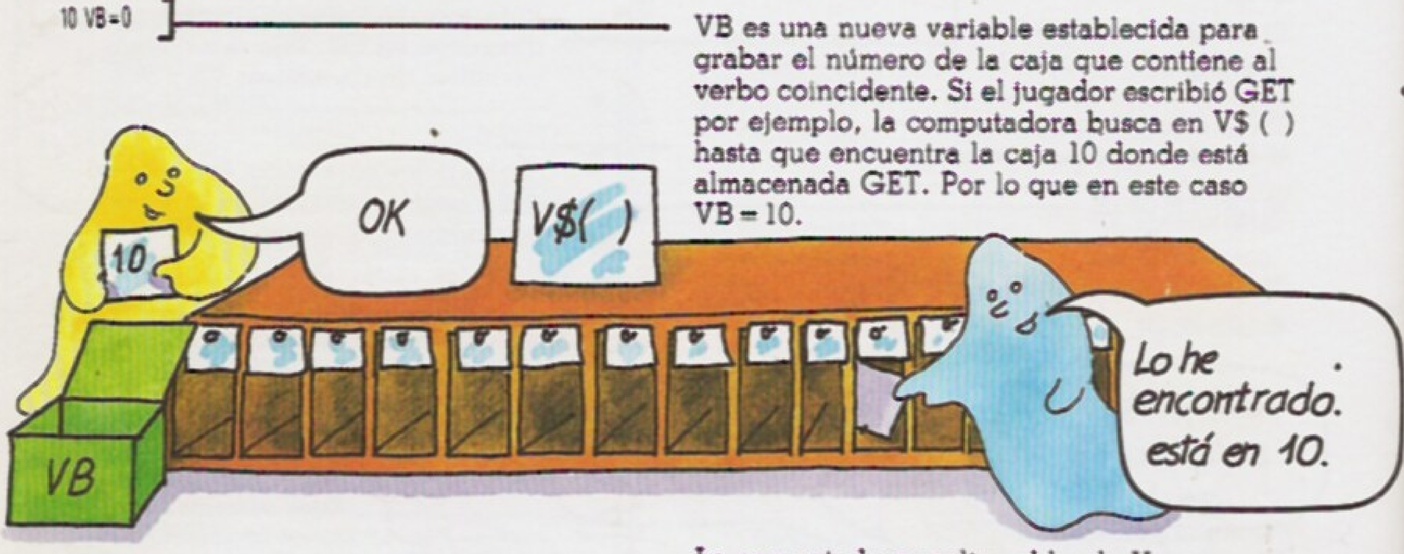
En este momento las instrucciones del jugador están almacenadas en las variables V$ y W$ que denominaremos «variable de verbos» y «variable de palabras», respectivamente. Lo siguiente será comparar éstas con las que introdujiste en el proceso de inicialización de la página 18. Asume que la palabra en V$ es un verbo y realiza un bucle para ver si coincide con cualquiera de los verbos en la matriz V$ ( ). (Observa la diferencia entre la variable de caracteres V$ y la matriz V$ ( ). Son totalmente diferentes para la computadora, por lo que debes tener cuidado de no confundirlos.)

De la misma manera la computadora ejecuta un bucle para ver si coincide W$ con alguna de las palabras de la matriz O$ ( ).

Esta es la parte del programa que comprueba la coincidencia entre las palabras del jugador y las palabras en la memoria de la computadora.

```
10 VB=0
```

VB es una nueva variable establecida para grabar el número de la caja que contiene al verbo coincidente. Si el jugador escribió GET por ejemplo, la computadora busca en V$ ( ) hasta que encuentra la caja 10 donde está almacenada GET. Por lo que en este caso VB = 10.




```
20 FOR I=1 TO V
30 IF V$=V$ (I) THEN VB=I
40 NEXT I
```

La computadora realiza el bucle V veces (V es el número de verbos en la memoria de la computadora) comparando el verbo del jugador con todos aquellos de su memoria. Si encuentra uno que coincide fijará el número correspondiente en VB.

```
50 OB=0
60 FOR I=1 TO W
70 IF W$=O$ (I) THEN OB=I
80 NEXT I
```

El bucle para W$ funciona de la misma manera, utilizando OB para grabar el número de la caja de la palabra coincidente.

¿Qué pasa si no coinciden las palabras?

Si no se encontrase ninguna coincidencia, entonces VB y OB seguirían siendo cero. La computadora entiende con esto como que la caja cero de la matriz contenía la palabra del jugador. Pero cuando busca allí la palabra coincidente no encuentra nada, ya que esta caja la dejaste vacía al rellenar las matrices con datos.



Combinaciones tontas

Observa que este proceso sólo comprueba si existen palabras coincidentes en la memoria de la computadora. No comprueba si la combinación de palabras tiene sentido. Una combinación tonta como ABRIR VELA es admitida en esta etapa del programa, pero será rechazada más adelante cuando la computadora intente llevar a cabo la acción. Es mucho más rápido hacer que en esta etapa la computadora sólo busque la coincidencia de palabras, no su validez.

Al final de esta sección del programa la computadora tiene un valor para VB y otro para OB. En la siguiente página verás lo que hace con éstos.

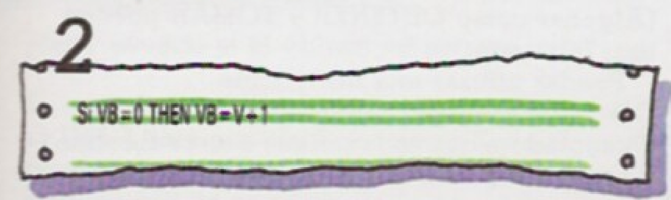
Mensajes de error

La computadora puede usar los valores VB y OB para ver si es necesario mandarle al jugador un mensaje que diga que sus instrucciones no son válidas. Esta parte del programa es como un filtro. Las instrucciones del jugador pasan una serie de tests. Si no pasan alguno de los tests se introduce un nuevo mensaje en M$; si superan todos los tests M$, todavía conserva el mensaje «¿QUE?», que se le asigna en el paso 220 (recuerda que los mensajes no aparecen todavía en la pantalla, sino que simplemente se asignan por lo que pueden cambiarse más adelante).

Estos son los pasos del programa que asignan los mensajes de error en la *Casa Encantada* —mira a ver si los encuentras en el listado principal. Necesitarás pasos semejantes si estás escribiendo tu propia aventura.



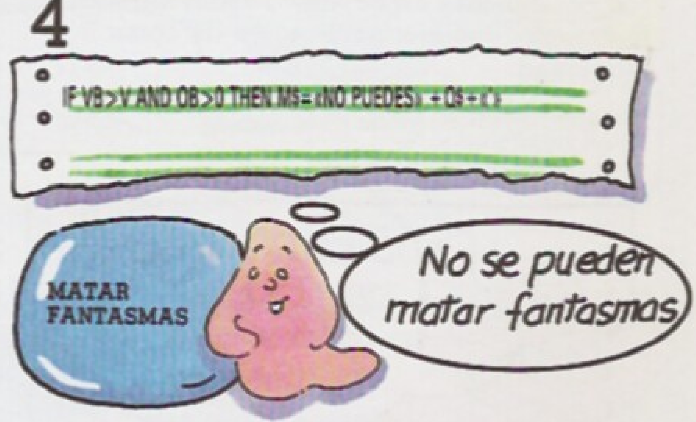
El primer test comprueba si hay una palabra en W$ (es decir, si el jugador escribió dos palabras) y a continuación comprueba si el valor de OB es cero.



Este paso es para evitar un problema. No puedes mandar un GOSUB con un valor cero. Pero las diferentes versiones de BASIC tienen maneras distintas de hacer esto. La mayoría ignoran el GOSUB y continúan con el siguiente paso. Sin embargo, algunas como el BBC, no admiten el cero y producen un mensaje de error. Para evitar esto a VB se le da un valor mayor que V (el número de verbos en la memoria de la computadora) y se manda a la computadora a un subrutina sin ningún valor.



Este paso manda un mensaje si el jugador escribió una sola palabra; es decir, si W$ está vacía. (Si la palabra es una de las instrucciones de una palabra permitida, este mensaje cambiará más adelante en el programa.)

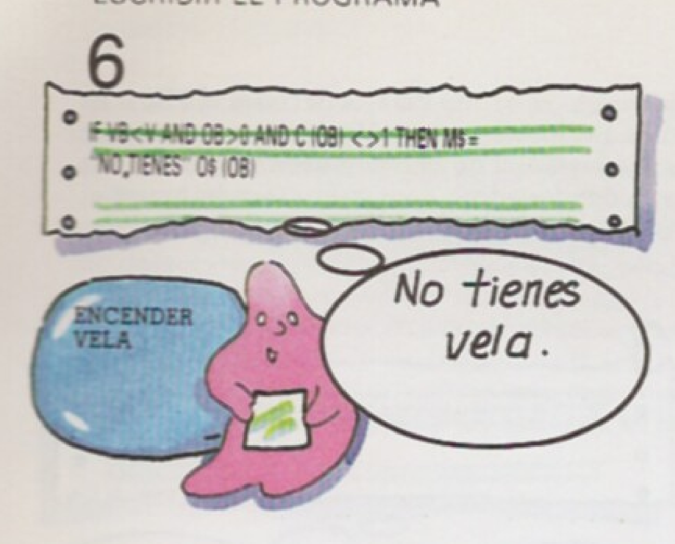


Este paso manda un mensaje si la computadora no tiene el verbo en su memoria pero sí tiene el objeto.



Si la computadora no tiene ninguna de las palabras del jugador en su memoria manda el siguiente mensaje.

6



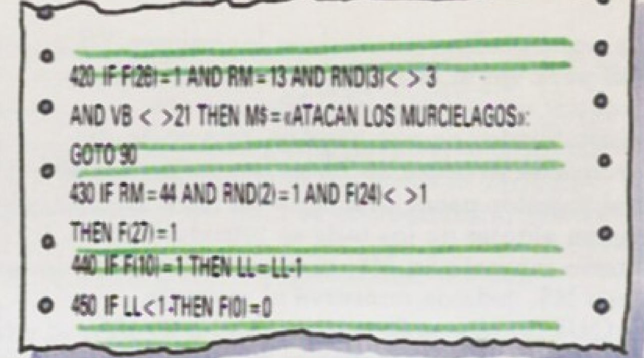
```
IF VB<V AND OB>0 AND C(OB)<>1 THEN M$=
"NO TIENES" O$(OB)
```


Condiciones especiales

En ocasiones suceden cosas en la aventura que impiden que el jugador continúe hasta que haya resuelto el problema. En esta situación aquellas instrucciones que normalmente serían válidas han de ser inutilizadas, para lo cual la computadora necesita algunos pasos para colocar banderas* en la memoria que le indiquen qué condiciones especiales deben aplicarse.

En la *Casa Encantada* los pasos 420 a 450 son condiciones especiales. Puedes verlas a la derecha con una explicación de cómo funcionan.





```
420 IF F(26)=1 AND RM=13 AND RND(3)< >3
AND VB < >21 THEN M$=«ATACAN LOS MURCIELAGOS»:
GOTO 90
430 IF RM=44 AND RND(2)=1 AND F(24)< >1
THEN F(27)=1
440 IF F(10)=1 THEN LL=LL-1
450 IF LL<1 THEN F(0)=0
```

Paso 420
Si hay murciélagos el jugador está en la habitación del torreón trasero, el número aleatorio no es 3 y el jugador no ha utilizado el verbo 21 (fumigar) en sus instrucciones; en ese caso, M$ se convierte en «Atacan los murciélagos», que impide que el jugador pueda continuar.

Paso 430
El jugador está en la habitación con telarañas, el valor del número aleatorio es 1 y la aspiradora está apagada. Esto hace que la bandera indique la aparición de fantasmas; es decir, F(27) pasa a valor 1.

Paso 440
Si la vela está encendida el contador de límite de luz, LL, va disminuyendo.

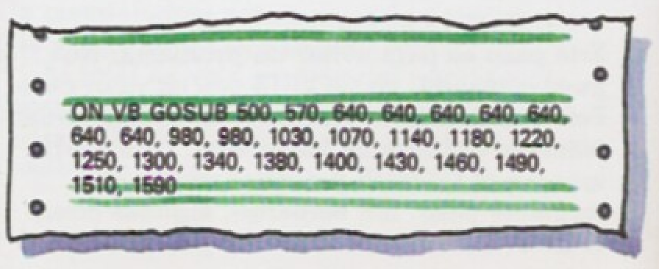
Paso 450
Si LL es cero, la bandera que hace que la vela esté encendida o apagada toma el valor cero.

Quizá se te ocurran otras condiciones especiales que puedas añadir a esta lista.

Ramificación a subrutinas

La siguiente labor de la computadora es realizar lo que le pide el jugador. Si tuviera que revisar todas las acciones para ver cuál es la requerida, el juego sería lento y pesado. Para evitarlo se utilizan multitud de subrutinas —casi una para cada verbo de la lista. (Algunas como OBTENER y TOMAR pueden usar la misma.)

Puedes utilizar una instrucción ON ... GOSUB para indicarle a la computadora que se ramifique a una subrutina diferente según el valor de VB.



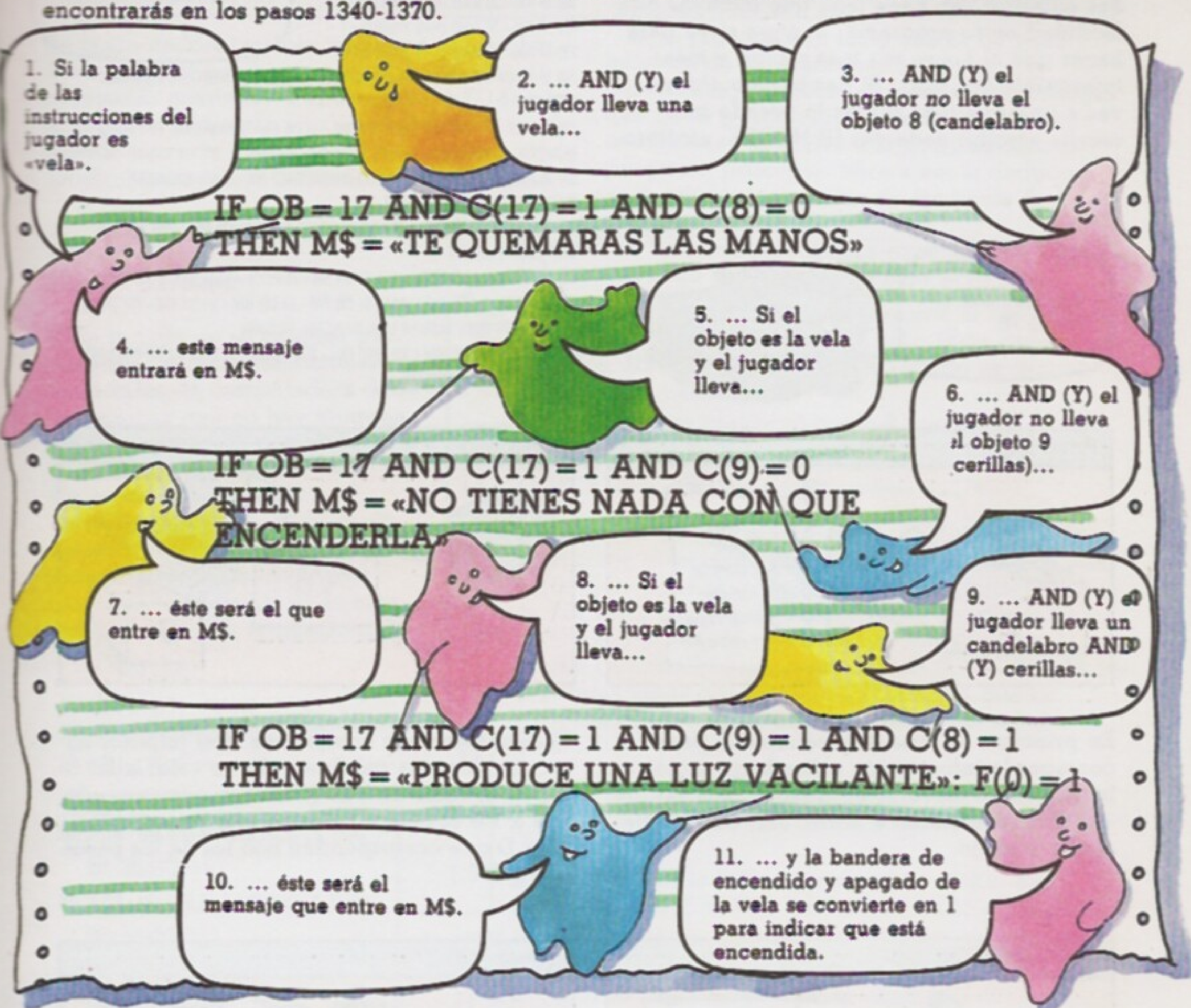
```
ON VB GOSUB 500, 570, 640, 640, 640, 640, 640,
640, 640, 980, 980, 1030, 1070, 1140, 1180, 1220,
1250, 1300, 1340, 1380, 1400, 1430, 1460, 1490,
1510, 1590
```


* **En la página 18 encontrarás más sobre banderas.**

Cómo funciona el paso ON ... GOSUB

El paso ON ... GOSUB de la página anterior funciona así. Si VB = 1, la computadora va al primer número de paso de la lista, si VB = 2 va al segundo, si VB = 3 va al tercer... Observa que el último número de paso de la lista es de una subrutina «inútil» para VB = V + 1 (el valor de VB cuando no se encuentra un verbo coincidente en la memoria de la computadora). Se le manda a un paso que simplemente dice RETURN; es decir, que devuelve a la computadora al programa principal.

Observa las subrutinas en las páginas 34 a 36 y mira si puedes sacar el trabajo con todo lo que ellos hacen. Aquí está el procedimiento de LIGHT (VB = 19) como un ejemplo. Lo encontrarás en los pasos 1340-1370.



¿Qué sucede si el objeto no es una vela?

Si el objeto que quiere utilizar el jugador no es VELA sino otro de la memoria de la computadora como PUERTA, entonces el mensaje M$ no se altera de lo que se le introdujo en el paso 220. Cuando la computadora vuelve al programa principal y encuentra la instrucción PRINT M$ escribirá el mensaje «¿QUE?».

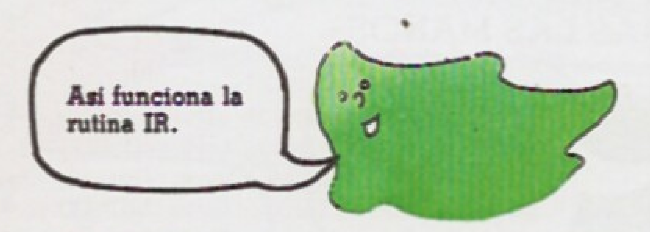
Observa que no hace falta elaborar un mensaje que diga que la vela no está ahí; se evita gracias a la sección de mensajes de error.

Vuelta al programa principal

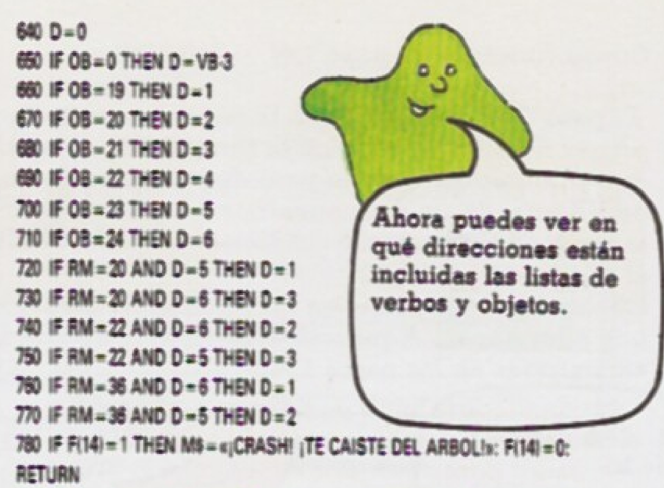
Aunque algunas de las rutinas de los verbos son más largas y complicadas que ésta, todas funcionan de la misma manera; el valor OB se comprueba, si es necesario la computadora manda un mensaje, y finalmente vuelve al programa principal. Comprueba los límites de la luz en los pasos 470 y 480 y vuelve al apartado de descripción y feedback. Cuando está aquí escribe el valor que introdujo en M$ y espera las siguientes instrucciones del jugador.

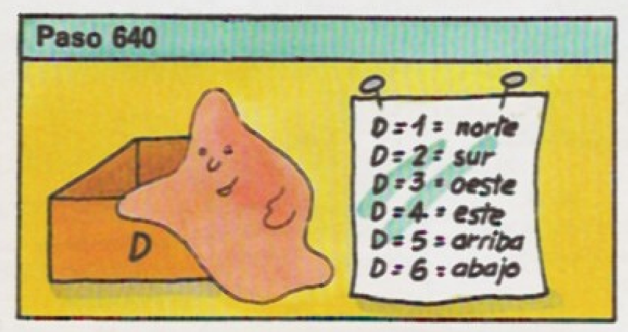
La subrutina IR

La subrutina para el verbo IR es tan amplia e importante en una aventura que casi podrías considerarla como un subprograma. Siete instrucciones verbales llevan a ella —IR, N, S, W, E, A y D—. Esta rutina también es especial porque responde a instrucciones de una sola letra, así como a instrucciones de dos palabras. No hace falta que incluyas esta facilidad en tu programa, aunque sirve para hacer que el juego sea más rápido y más interesante de jugar. Si has jugado alguna vez a una aventura sabrás lo pesado que resulta escribir cada vez IR NORTE, etcétera.

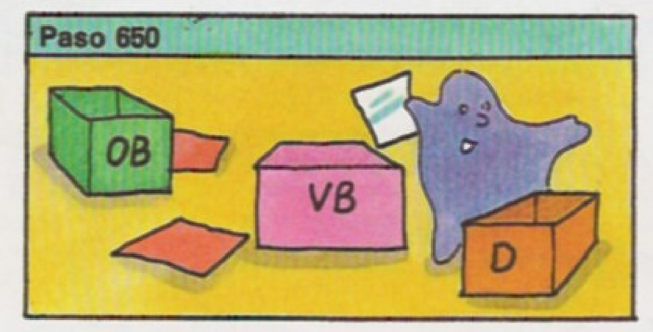



```
640 D=0
650 IF OB=0 THEN D=VB-3
660 IF OB=19 THEN D=1
670 IF OB=20 THEN D=2
680 IF OB=21 THEN D=3
690 IF OB=22 THEN D=4
700 IF OB=23 THEN D=5
710 IF OB=24 THEN D=6
720 IF RM=20 AND D=5 THEN D=1
730 IF RM=20 AND D=6 THEN D=3
740 IF RM=22 AND D=6 THEN D=2
750 IF RM=22 AND D=5 THEN D=3
760 IF RM=36 AND D=6 THEN D=1
770 IF RM=36 AND D=5 THEN D=2
780 IF F(14)=1 THEN M$=«¡CRASH! ¡TE CAISTE DEL ARBOL!»: F(14)=0:
RETURN
790 IF F(27)=1 AND RM=52 THEN M$=«LOS FANTASMAS NO TE DEJAN
MOVERTE»: RETURN
800 IF RM=45 AND C(1)=1 AND F(34)=0 THEN M$=«UNA BARRERA
MAGICA AL OESTE»: RETURN
810 IF (RM=26 AND F(0)=0) AND (D=1 OR D=4) THEN M$=«NECESITAS
UNA LUZ»: RETURN
820 IF RM=54 AND C(15) <>1 THEN M$=«ESTAS ATASCADO»: RETURN
830 IF C(15)=1 AND NOT (RM=53 OR RM=54 OR RM=55 OR RM=47) THEN
M$=«¡NO PUEDES LLEVAR UNA BARCA!»: RETURN
840 IF (RM>26 AND RM<30) AND F(0)=0 THEN M$=«DEMASIADO OSCURO
PARA MOVERSE»: RETURN
```

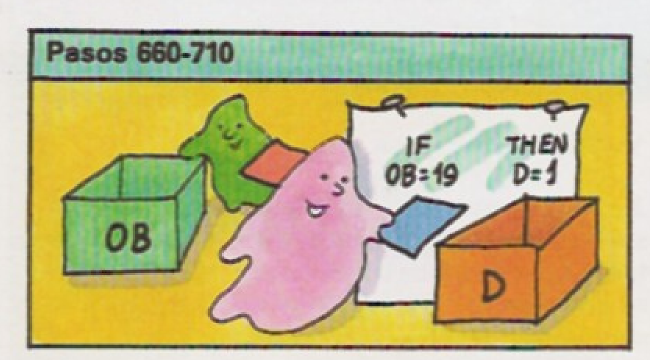




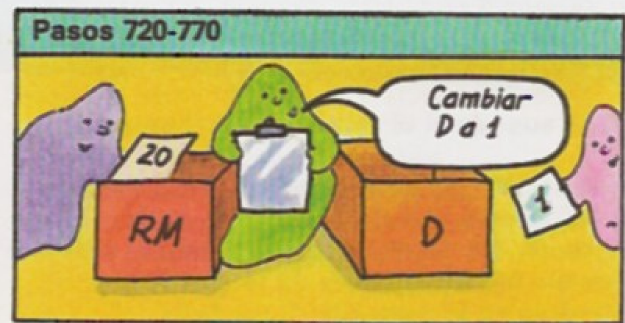
En principio se crea una variable D para contener la información sobre la dirección en la que desea moverse el jugador. Los valores 1 al 6 corresponden a norte, sur, oeste, este, arriba y abajo.



El siguiente paso comprueba si el jugador ha escrito sólo una palabra y da un valor a D según el valor de VB. (Observa que restando tres a VB la computadora obtiene resultados para D que corresponden con los de los pasos 660 a 710.)



Los siguientes seis pasos comprueban si el jugador escribió una instrucción de dos palabras. Usa el valor de OB para establecer el de D.



Al no ser ésta una aventura en tres dimensiones las instrucciones ARRIBA y ABAJO han de convertirse en norte, sur, este u oeste. Los pasos 720 a 770 hacen esto (si revisas el plano principal de las páginas 14 y 15 observarás cómo funciona esto).

Pasos 780-840



La computadora también tiene que comprobar si hay alguna condición especial que impida al jugador continuar. Por ejemplo, si F(14) = 1 el jugador está en lo alto de un árbol. Si intenta moverse sin bajar primero del árbol recibirá un mensaje diciendo que se ha caído.

Si el jugador está en la casilla 52 y la bandera de los fantasmas está «on» (encendida) se manda un mensaje indicándole que no se puede mover. Cada una de estas condiciones devuelve a la computadora al programa principal. Mira a ver si deduces para qué sirven el resto de los pasos de esta sección.

Buscar paredes

Si la jugada no se bloquea por ninguna de estas condiciones especiales, la computadora debe comprobar que no hay ninguna pared ni ningún obstáculo bloqueando el camino. Estos son los pasos que hacen esto. A primera vista parecen bastante complicados, pero si los miras con atención, recordando lo que significa cada variable, te será posible entender lo que sucede.

```
850 F(35)=0:RL=LEN (R$(RM))
860 FOR I=1 TO RL
870 U$=MID$ (R$(RM), I, 1)
880 IF (U$=«N» AND D=1 AND F(35)=0: THEN
    RM=RM-8:F(35)=1
890 IF (U$=«S» AND D=2 AND F(35)=0: THEN
    RM=RM+8:F(35)=1
900 IF (U$=«W» AND D=3 AND F(35)=0: THEN
    RM=RM-1:F(35)=1
910 IF (U$=«E» AND D=4 AND F(35)=0: THEN
    RM=RM+1:F(35)=1
920 NEXT I
930 M$=«OK»
940 IF F(35)=0 THEN M$=«IMPOSIBLE IR POR ESE CAMINO»
950 IF D<1 THEN M$=«¿IR DONDE?»
960 IF RM=41 AND F(23)=1 THEN R$(49)=«SW»:M$ - «LA
    PUERTA SE CIERRA DANDO UN PORTAZO»: F(23)=0
970 RETURN
```

Esta es una bandera para que la computadora registre si ha encontrado la salida que quiere el jugador.

RL es una nueva variable que contiene la longitud de la cadena que encuentra en R$(RM). (Esta cadena son las rutas, NSW, etcétera, para el terreno en que se encuentra el jugador.)

La computadora realiza el bucle RL veces.

Cada vez la computadora quita uno de los caracteres en R$(RM) y lo domina temporalmente U$.

A continuación realiza unos tests con U$ y D. Si la instrucción direccional del jugador coincide con una salida que existe en el terreno donde se halla, el valor de RM varía para trasladarle al lugar correspondiente. F(35) toma el valor 1 para impedir que la computadora intente cambiar RM de nuevo al continuar el bucle. (Si lo piensas verás que esto sería posible ya que la computadora utiliza su nuevo valor de RM en el paso 870.)

Si revisas el plano principal verás que sumando o restando 1 u 8 el jugador se mueve al siguiente terreno correcto.

Al final del bucle M$ se convierte en «OK». Esto sustituirá a «NECESITO DOS PALABRAS» que fue introducido como mensaje de error por si el jugador escribía una instrucción de una sola palabra.

Si F(35) es todavía cero quiere decir que el desplazamiento que quería realizar el jugador no está permitido, por lo que M$ varía para indicar esto.

Si D es menor que uno (es decir, que no se le asigna un valor en los pasos 650 a 770), entonces M$ se convierte en «¿IR DONDE?»

Este paso convierte la puerta principal en un camino de una sola dirección. Cuando el jugador entra en el terreno 41 (el recibidor), las salidas del terreno 49 (porche delantero) se cambian de «NSW» a «SW», M$ se convierte en «LA PUERTA SE CIERRA CON UN PORTAZO», y la bandera de la puerta principal toma el valor cero para indicar a la computadora que en ese momento está cerrada. (Las salidas del terreno 41 no necesitan cambiarse, ya que S no estaba incluido desde un principio.)

Cambiar el programa

Puedes hacer todos los cambios que quieras en el programa de este libro, ya sea para hacer variaciones en el tema de la *Casa Encantada* o bien para crear juegos con ambientaciones, descripciones, objetos, verbos y mensajes totalmente diferentes. Recuerda que cuantos más cambios hagas más difícil te resultará, ya que tendrás que comprobar cómo cada cambio afecta al resto del juego.

Si vas a escribir un nuevo programa, utilizando éste como guía, deberás planearlo como se explica en las páginas 6 a 15. Merece la pena desperdiciar algo de tiempo planeando con cuidado el juego, ya que de esta forma no encontrarás fallos a la hora de realizarlo.

Es una buena idea comenzar haciendo pequeños cambios para ver lo que sucede. Si guardas el juego original en una cinta, puedes hacer cambios, probarlos y adaptarlos sin perder el original.

¿Cuánta memoria extra tienes?



La *Casa Encantada* ocupa alrededor de 7 K. de memoria RAM antes de ser ejecutada. Luego necesitará de 3½ a 4 K. más para que las matrices almacenen los datos. Tu computadora absorberá algo de memoria para uso interno —hasta 3 K. en algunos modelos—, así como 1 K. o más para la pantalla. (El spectrum utiliza 7 K., por lo que la *Casa Encantada* no vale en el modelo de 16 K.) Por ello, si tienes una computadora de 16 K., no tendrás mucha memoria extra, por lo cual las alteraciones tendrán que ser cambios no añadidos.

Descripciones más largas

Si tienes más de 16 K., una de las maneras más fáciles de hacer el juego más interesante es añadir descripciones más largas. En lugar de «impresionante vestíbulo abovedado» podrías decir algo como «Has penetrado en un vestíbulo abovedado inmenso, con columnas de fantástica altura. La luz se filtra del este y gracias a ello se vislumbra; por el oeste, algo que podría ser una puerta...».

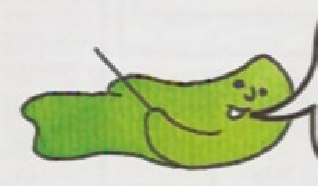


Añade un límite de tiempo

La *Casa Encantada* ya posee un límite de tiempo en la duración de la vela encendida. Puedes añadir otros límites de tiempo, como contar el número de veces que el jugador se ha movido y hacer que la computadora pare el juego cuando llegue a un número determinado.

Puedes hacer esto añadiendo al paso 70 y poniendo un paso extra en 485.

```
70 V=25:W=36:G=18:T=0
```



Quizá no tengas que hacer esto, ya que no todas las computadoras necesitan que se establezcan nuevas variables antes de utilizarlas.

```
485 T=T+1:IF T>200 THEN PRINT «HA CAIDO LA
NOCHE. TE HAS CONVERTIDO EN UN MURCIELAGO»:
STOP
```

Puedes cambiar este número por cualquier otro.

Problema

¿Se te ocurre cómo poner un límite al número de objetos que el jugador puede llevar al mismo tiempo? (Tendrás que ajustar una rutina para que los sume.)

Añadir sonidos



Este es un método eficaz para añadirlos al juego sin realizar cambios complicados. Necesitarás saber cómo funcionan las instrucciones de sonido de tu computadora. Prueba primero algunas rutinas de sonido para encontrar la que quieres y luego añade una instrucción GOSUB al paso donde sucede la acción, ej.: IF RM = 46 AND C(1) = 1 THEN M$ = «ALGO EXTRAÑO ESTA SUCEDIENDO» GOSUB 6000.

Puedes añadir sonidos para el portazo, para la destrucción de la puerta secreta, para la llave al abrir la puerta, para los efectos mágicos al decir la palabra mágica, etcétera. Esta tabla contiene algunas rutinas de sonidos para variar computadoras. El único límite al añadirlas es la cantidad de memoria.

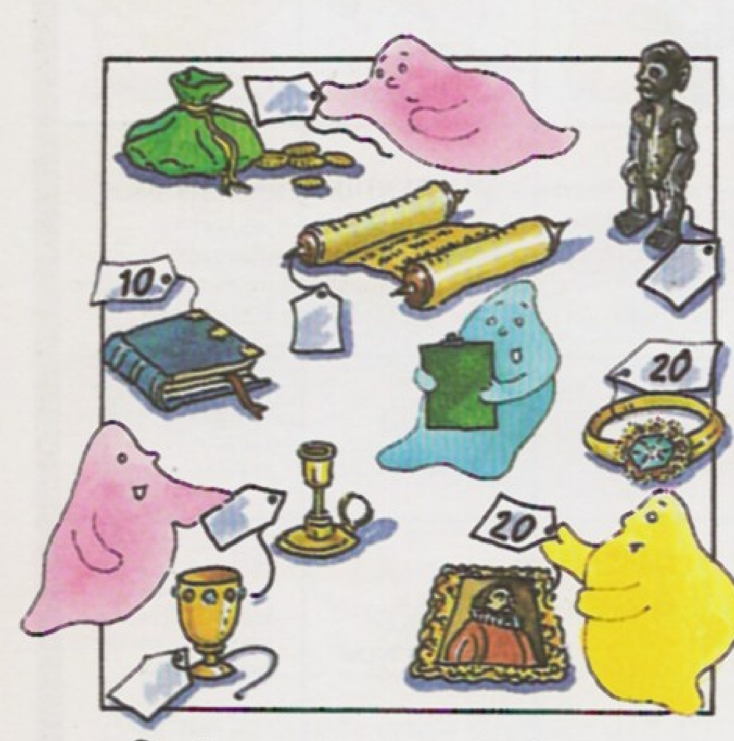
	CAERSE DEL ARBOL	PORTAZO	MAGIA	HACHAZO	SUCESO GENERAL
VIC 20	POKE 36877,130 FOR L = 15 TO O STEP −1 POKE 36878,L FOR M − 1 TO 20: NEXT M NEXT L POKE 36877,0	POKE 36877,130 FOR L = 15 TO O STEP −1 POKE 36878,L FOR M = 1 TO 20:NEXT M NEXT L POKE 36877,0	POKE 36878,15 FOR I = 160 TO 240 STEP 5 POKE 36876,I FOR M = 1 TO 100:NEXT M POKE 36876,0	POKE 36878,15 FOR I = 1 TO 10 POKE 36877,200 POKE 36877,0 FOR M = 1 TO 400:NEXT h NEXT I	POKE 36878,15 FOR = 1 TO 2 POKE 36876,200 POKE 36876,0 FOR M = 1 TO 400:NEXT M NEXT I
SPECTRUM	—	—	FOR I = 5 TO 40 STEP 3 BEEP 0.2,1 NEXT I	FOR I − 1 TO 10 BEEP 0.01, 0.01 FOR M = 1 TO 100:NEXT M NEXT I	BEEP 0.5,5 PAUSE 50 BEEP 0.5,5
BBC	FOR L = −15 TO 0 SOUND 0,L,5,1 NEXT L	FOR L = −15 TO −B SOUND 0,L,5,0.6 NEXT L	FOR I − 40 TO 160 STEP 5 SOUND 2, −15,I,5 NEXT I	FOR I = 1 TO 10 SOUND 0, −15,5,1 FOR M = 1 TO 1000:NEXT M NEXT I	SOUND 2, −15,100,2 FOR M − 1 TO 1000:NEXT M SOUND 2, −15,100,2
DRAGON/ TRS-COLOR	—	—	FOR 1 − 50 TO 230 STEP 10 SOUND I,2 NEXT I	—	SOUND 180.1 FOR M = 1 TO 500:NEXT M SOUND 180,1
ORIC	FOR L = 15 TO O STEP −1 PLAY 0,2,6,100 SOUND 5,100,L WAIT 1 NEXT L PLAY 0,0,0,0	FOR L = 15 TO O STEP −I PLAY 0,2,6,100 SOUND 5,100,L WAIT 1 NEXT L PLAY 0,0,0,0	FOR L − 15 TO O STEP −1 PLAY 0,2,6,100 SOUND 5,100,L WAIT 10 NEXT L PLAY 0,0,0,0	FOR I − 1 TO 10 PLAY 0,1,1,20 SOUND 4,50,7 PLAY 0,0,0,0 FOR M = 1 TO 300:NEXT M NEXT I	FOR I = 1 TO 2 PLAY 1,0,6,100 SOUND 1,70,7 WAIT 20 PLAY 0,0,0,0 FOR M − 1 TO 300:NEXT M NEXT I

Puntuación

La *Casa Encantada* tiene un sistema de puntuación muy sencillo, dando un punto por cada objeto que lleva el jugador. Puedes cambiarlo y poner un sistema más interesante que se base en que cada objeto tenga un valor diferente. Si asumes que los objetos están numerados en orden descendente de valor, entonces el cuadro será el más valioso y la llave el menos valioso. Si cambias el paso 1530 y pones:

```
1530 IF C(I)=1 THEN S=S+G—I
```

entonces el cuadro valdrá 18—1=17 y la llave 18—18=0 (G es el número de objetos que se pueden obtener e I el número de objetos que lleva el jugador). Esto hace que la llave no tenga ningún valor como tesoro, aunque sigue siendo un objeto muy valioso por su utilidad, ya que sin ella el jugador no podrá conseguir ni la copa ni el cuadro.



Si quieres un sistema más flexible (y tienes suficiente memoria) puedes establecer una matriz que contenga los valores de los objetos en la rutina de inicialización como ésta:




```
DATA 20, 20, 30, 11, 16, 25, 32, 8, 25, 4, 9, 17, 3, 0,
10, 12, 4, 9
```

Añade los números de paso.

Dimensiona la nueva matriz T con G espacios (es decir, con el número de objetos que se pueden recoger).

```
DIM T(G)
FOR I=1 TO G
READ T(I)
NEXT I
```

Cambia también el paso 1530 a:
1530 IF C(I)=1 then S=S+T(I).

¡Qué tonto!

Castigar

MK

De momento la puntuación se ha basado en puntos a favor y no la han afectado las tonterías que puede cometer el jugador. Podrías añadir un sistema de castigo utilizandó un contador como puede ser MK, para los errores.
Cuando el jugador haga algo realmente tonto, sumar algo a MK y luego lo sustraes de S cuando se calcula la puntuación en el paso 1530. Si el jugador se cae del árbol, por ejemplo, puedes sumar un punto (o más) de castigo de la siguiente forma:

```
780 IF F(14)=1 THEN M$=«¡CRASH! TE CAISTE DEL ARBOL»: F(14)=0:
MK=MK+1: RETURN
```

No te olvides que algunas computadoras necesitan que se creen las variables antes de utilizarlas. Puedes hacer esto añadiendo MK=O a las variables del paso 70.

Grabar el juego



Sería interesante poder apagar a mitad del juego y luego continuar desde donde lo dejaste. Con juegos largos y complicados es un factor importante el hacer esto añadiendo SAVE y LOAD a la lista de verbos. En el paso 70 cambia el valor de V a 27 y añade dos nuevos verbos separados con comas al final del paso 1665. También tendrás que cambiar el paso ON GOSUB número 460.

Pon el número de los pasos de las dos nuevas subrutinas (una para SAVE y otra para LOAD) entre los dos últimos números del paso 460 para que se tenga:
... 1510, 3000, 4000, 1590.

Primer número nuevo — Segundo número nuevo

Luego añade las nuevas subrutinas comprobando el manual de tu computadora para asegurarte de que las palabras son correctas.

```
• 3000 INPUT «ESTA TU
CASSETTE LISTO PARA
GRABAR»; Y $
3010 IF Y $<>«Y» THEN
3000
3020 OPEN FILE FOR
OUTPUT FROM COMPUTER
3030 PRINT #1, RM
```

Sustituye este paso por el que requiera tu computadora. Quizá no necesites ningún paso.

Esto graba el terreno en el que se halla el jugador.

Este bucle graba las posiciones de objetos que se pueden tomar.

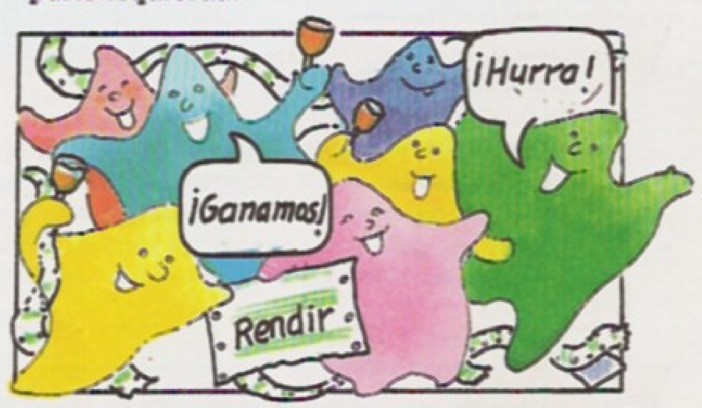
```
3040 FOR I=1 TO G
3050 PRINT #1, L(I)
3060 NEXT I
3070 FOR I=1 TO W
3080 PRINT #1, C(I), F(I)
3090 NEXT I
3100 CLOSE
3200 RETURN
• 4000 INPUT «¿ESTAS LISTO PARA CARGAR»; Y $
4010 IF Y $ <>«Y» THEN 4000
4020 OPEN FILE FOR INPUT TO COMPUTER
4030 INPUT #1, RM
4040 FOR I=1 TO G
4050 INPUT #1, L(I)
4060 NEXT I
4070 FOR I=1 TO W
4080 INPUT #1, C(I), F(I)
4090 NEXT I
4100 CLOSE
4200 RETURN
```

Esta graba los objetos que lleva el jugador y el estado de las banderas.

Observa que esta rutina para grabar no graba las descripciones y las salidas del juego. Esta significa que las habitaciones y las salidas alteradas por la acción del jugador vuelven a su estado inicial —la pared secreta se reconstruirá, la puerta se volverá a cerrar, etcétera. (Habrá que echar la culpa a los fantasmas.) Podrías grabar las matrices D$ y R$ si quisieras, añadiendo nuevos bucles a las rutinas de SAVE y LOAD.

¿Te rindes?

Como casi todas las aventuras, la *Casa Encantada* contiene trampas para los jugadores, de las que sólo pueden salvarse utilizando un determinado objeto de una forma determinada. Si el jugador no tiene ese objeto se queda atascado. Las posibilidades de «rendirse» serían útiles en estos casos para que el jugador no tenga que apretar BREAK o ESCAPE para terminar el juego. Pueden hacer esto añadiendo RENDIR a la lista de verbos y elaborando una nueva subrutina como las de SAVE y LOAD descritas en la parte izquierda.



Acuérdate de cambiar el valor de V en el paso 70, de añadir RENDIR al final del paso 1665 y de introducir el paso de la nueva subrutina en el paso 460, situándolo en segundo lugar si se comienza a contar por detrás.

La subrutina de RENDIR sería algo así:

```
5000 INPUT «DESEA RENDIRSE»; Q$
5010 IF Q$<>«Y» THEN RETURN
5020 INPUT «QUIERE GRABAR PRIMERO EL JUEGO»; Q$
5030 IF Q$=«Y» THEN GOSUB 3000
```

No necesitarás esto si no has incluido la subrutina para grabar.

```
5040 PRINT «GRACIAS POR JUGAR»
5050 END
```

Observa que al final de esta subrutina no hay un RETURN. Esto está contra las reglas del BASIC, pero en este caso la computadora no puede confundirse, ya que al llegar al paso 5050 el programa dejará de ejecutarse.

• Si tienes un BBC tendrás que sustituir el punto y coma por una coma.

Eliminación de los posibles errores

Si has elaborado tu propia versión de la *Casa Encantada* o añadido las rutinas de complicación, lo más probable es que hayas cometido errores. Estos son algunos de los problemas con los que te puedes encontrar, así como algunas soluciones para corregirlos.

Out of data

Si la computadora te indica un error del tipo «out of data in line X» (datos incorrectos en el paso X), significa que en una de tus secciones de lectura de datos se leen datos incorrectos o incompletos. Puedes haber olvidado una coma en la instrucción DATA, o bien un dato, o incluso haber puesto un número erróneo en el bucle.

```
DIM A(4)
FOR I=1 TO 4
READ A(I)
NEXT I
DATA ESPADA, DINERO, COMIDA, AGUA
```

Falta una coma aquí

Error aquí

Array error

Si obtienes un array error (error en la matriz) significa que no reservaste suficiente espacio al DIMensionar la matriz, o bien que pusiste un elemento de más en la instrucción DATA (quizá poniendo una coma de más) y luego lo contaste al calcular el número para el bucle de lectura: READ.

```
DIM F(3)
FOR I=1 TO 4
READ A(I)
NEXT I
DATA HACHA, ATAUD, SANGRE, LLAVE
```

Los objetos se comportan de forma extraña

Esto puede suceder porque el programa esté mal bifurcado y mediante el paso ON GOSUB se dirija a una subrutina incorrecta y no a la que debiera...

Comprueba todos los números de este paso con las subrutinas con el mismo número. Si está todo bien, comprueba las instrucciones DATA de los verbos para ver si su orden coincide con el orden de las subrutinas.

Si el programa se dirige a una subrutina de forma correcta y los verbos están en el orden preciso, comprueba que hay un RETURN al final de cada subrutina. Si falta la computadora seguirá ejecutando una subrutina después de la otra, produciendo extraños resultados.

Si nada de esto te resuelve el problema, revisa las condiciones de subrutinas. Puedes haberte olvidado algo o haber puesto mal un signo o haber utilizado una variable donde no debías. Comprueba las condiciones especiales y las banderas que introducen al principio del programa.

Salidas en sitios curiosos

Si encuentras una pared que puedes atravesar o una puerta que no, es probable que hayas cometido algún fallo al planear las salidas o al introducir los datos de las salidas. Revisa tu mapa de salidas con los pasos de datos para encontrar el error.

Los objetos no aparecen donde debieran

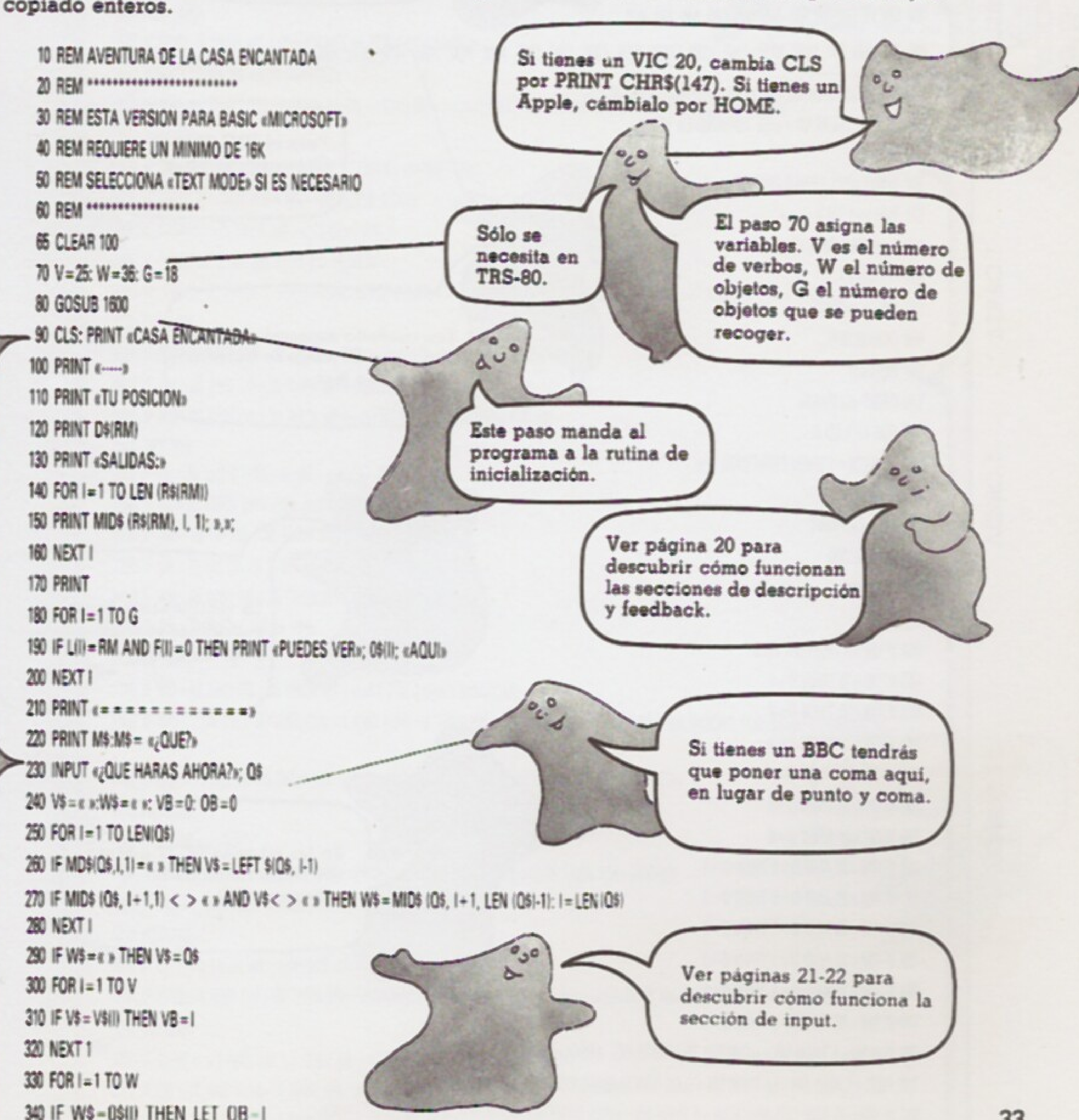
Si algún objeto aparece en un lugar que no es el que le corresponde, es posible que hayas cometido un error en los datos para la matriz L. Si algún objeto ni siquiera aparece, revisa la matriz de banderas. Quizá hayas asignado el valor 1 a la bandera de ese objeto, lo que significa que el objeto está ahí, pero la computadora no te lo dice. Tendrás que poner a cero esa bandera. Revisa la rutina de inicialización donde se determinan las banderas y a continuación las referencias a las banderas que se hacen a lo largo de todo el programa.

Listado de la *Casa Encantada*

Este es el listado para la aventura de la *Casa Encantada*. Debe funcionar en cualquier computadora que utilice BASIC de tipo Microsoft y que tenga un mínimo de 16K de RAM. Quizá tengas que hacer algunos pequeños cambios para tu computadora —para ello observa los comentarios que hay junto a algunos pasos del listado—. Si tienes un VIC 20 o un Oric, tendrás que añadir algunos pasos y hacer algunos cambios que encontrarás al final del listado en la página 37. Si tienes un BBC modelo A utiliza el módulo 7.

Tal y como está el listado no funcionará en la computadora Sinclair (Timex). Si tienes un Spectrum (Timex 2000), pasa a la página 38 para ver los cambios que tienes que hacer al programa. Si tienes una ZX81 (Timex 1000) encontrarás un listado especial en las páginas 39 a 45.

Al ser éste un programa largo, tendrás que tener mucho cuidado al copiarlo. El mínimo error hará que no funcione correctamente y te será difícil encontrarlo una vez que hayas copiado todo el programa. Revisa cada paso según lo copias, especialmente los que contienen ON GOSUB y DATA. Algunos pasos son tan largos que ocupan dos o más líneas de página impresa. Fíjate en éstos y ten cuidado de no apretar RETURN o ENTER hasta que los hayas copiado enteros.




```
10 REM AVENTURA DE LA CASA ENCANTADA
20 REM ***********************
30 REM ESTA VERSION PARA BASIC «MICROSOFT»
40 REM REQUIERE UN MINIMO DE 16K
50 REM SELECCIONA «TEXT MODE» SI ES NECESARIO
60 REM *****************
65 CLEAR 100
70 V=25: W=36: G=18
80 GOSUB 1600
90 CLS: PRINT «CASA ENCANTADA»
100 PRINT «-----»
110 PRINT «TU POSICION»
120 PRINT D$(RM)
130 PRINT «SALIDAS:»
140 FOR I=1 TO LEN (R$(RM))
150 PRINT MID$ (R$(RM), I, 1); »,«;
160 NEXT I
170 PRINT
180 FOR I=1 TO G
190 IF L(I)=RM AND F(I)=0 THEN PRINT «PUEDES VER»; O$(I); «AQUI»
200 NEXT I
210 PRINT «==============»
220 PRINT M$:M$= «¿QUE?»
230 INPUT «¿QUE HARAS AHORA?»; Q$
240 V$=« »:W$=« »: VB=0: OB=0
250 FOR I=1 TO LEN(Q$)
260 IF MD$(Q$,I,1)=« » THEN V$=LEFT $(Q$, I-1)
270 IF MID$ (Q$, I+1,1) < > « » AND V$< > « » THEN W$=MID$ (Q$, I+1, LEN (Q$)-1): I=LEN (Q$)
280 NEXT I
290 IF W$=« » THEN V$=Q$
300 FOR I=1 TO V
310 IF V$=V$(I) THEN VB=I
320 NEXT 1
330 FOR I=1 TO W
340 IF W$=O$(I) THEN LET OB=I
```

Ver páginas 23-24 para averiguar cómo funciona la sección de errores.

MENSAJES DE ERROR Y CONDICIONES ESPECIALES

```
350 NEXT I
360 IF W$>« » AND OB=0 THEN M$=«ESO ES RIDICULO»
370 IF VB=0 THEN VB=V+1
380 IF W$=« » THEN M$=«NECESITO DOS PALABRAS»
390 IF VB>V AND OB> THEN M$=«NO PUEDES»+Q$+«'»
400 IF VB>V AND OB=0 THEN M$=«NO TIENE SENTIDO»
410 IF VB<V AND OB> AND C(OB)=0 THEN M$=«NO TIENES»+W$
420 IF F(26)=1 AND RM=13 AND RND(3)<>3 AND VB<>21 THEN M$=«ATACAN LOS MURCIELAGOS»: GOTO 90
430 IF RM=44 AND RND(2)=1 AND F(24)< >1 THEN F(27)=1
440 IF F(0)=1 THEN LL=LL-1
450 IF LL<1 THEN F(0)=0
```

Usa las instrucciones de tu computadora para RND.

BIFURCACION A SUBRUTINAS

```
460 ON VB GOSUB 500, 570, 640, 640, 640, 640, 640,
640, 640, 980, 980, 1030, 1070, 1140, 1180, 11220, 1250, 1300, 1340, 1380, 1400, 1430, 1460, 1490, 1510, 1590
```

Para el VIC 20 y el Oric cambia estos pasos. Ver página 37.

```
470 IF LL=10 THEN M$=«¡TU VELA SE VA GASTANDO!»
480 IFLL=1 THEM M$=«¡SE TERMINO LA VELA!»
490 GOTO90
```

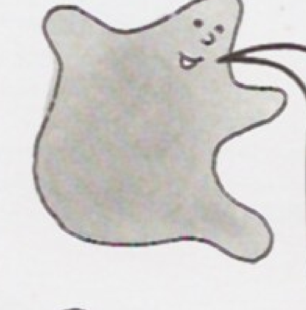
VERBO 1

```
500 PRINT «PALABRAS QUE SE»
510 FOR I=1 TO V
520     PRINT V$(I); «,»;
530     NEXT I
540 M$=« »:PRINT
550 GOSUB1580
560 RETURN
```



VERBO 2

```
570 PRINT «LLEVAS»
580 FOR I=1 TO G
590     IFC(I)=1 THEN PRINTO$(I); «,»;
600     NEXT I
610 M$=« »:PRINT
620 GOSUB1580
630 RETURN
```



VERBOS 3-9

```
640 D=0
650 IF OB=0 THEN D=VB-3
660 IF OB=19 THEN D=1
670 IF OB=20 THEN D=2
680 IF OB=21 THEN D=3
690 IF OB=22 THEN D=4
700 IF OB=23 THEN D=5
710 IF OB=24 THEN D=6
720 IF RM=20 AND D=5 THEN D=1
730 IF RM=20 AND D=6 THEN D=3
740 IF RM=22 AND D=6 THEN D=2
750 IF RM=22 AND D=5 THEN D=3
760 IF RM=36 AND D=6 THEN D=1
770 IF RM=36 AND D=5 THEN D=2
780 IF F(14)=1 THEN M$=«¡CRASH! TE CAISTE DEL ARBOL»: F(14)=0:RETURN
790 F(27)=1 AND RM=52 THEN M$=«LOS FANTASMAS NO TE DEJAN MOVERTE»:RETURN
800 IF RM=45 AND C(1) AND F(34)=0 THEN M$=«UNA BARRERA MAGICA POR EL OESTE»:RETURN
810 IF (RM=26 AND F(0)=0) AND (D=1 OR D=4) THEN M$«NECESITAS UNA LUZ»:RETURN
```



```
820 IF RM=54 AND C(15) <>1 THEN M$=«¡ESTAS ATASCADO!»:RETURN
830 IF C(15)=1 AND NOT (RM=53 OR RM=54 OR RM=55 OR RM=47)THEN M$=«¡NO PUEDES LLEVAR UNA BARCA!»:RETURN
840 IF (RM>26 AND RM<30) AND F(0)=0 THEN M$=«DEMASIADO OSCURO PARA MOVERTE»:RETURN
850=F(35)=0:RL=LEN (R$(RM))
860 FOR I=1 TO RL
870     U$=MID$(R$(RM),I,1)
880     IF (U$=«N» AND D=1 AND F(35)=0) THEN RM=RM-8:F(35)=1
890     IF (U$=«S» AND D=2 AND F(35)=0) THEN RM=RM+8:F(35)=1
900     IF (U$=«W» AND D=3 AND F(35)=0) THEN RM=RM-1:F(35)=1
910     IF (U$=«E» AND D=4 AND F(35)=0) THEN RM=RM+1:F(35)=1
920     NEXT I
930 M$=«OK»
940 IF F(35)=0 THEN M$=«IMPOSIBLE IR POR ESE CAMINO»
950 IF D<1 THEN M$=«¿IR DONDE?»
960 IF RM=41 AND F(23)=1 THEN R$(49)=«SW»:M$=«¡LA PUERTE SE CIERRA DANDO UN PORTAZO!»:F(23)=0
970 RETURN
```

Si tienes un VIC 20 podrías acotar el mensaje a 22 caracteres o asegurarte de que los espacios quedan situados de tal manera que queden bien en la pantalla.

VERBOS 10 Y 11

```
980 IF OB>G THEN M$=«NO PUEDES COGER»+W$RETURN
985 IF L(OB) <> RM THEN M$=«NO ESTA AQUI»
990 IF F(OB)< >0 THEN M$=«¿QUE?»+W$«?»
1000 IF C(OB)=1 THEN M$=«YA LO TIENES»
1010 IF OB>0 AND L (OB)=RM AND F(OB)=0 THEN C (OB)=1:L(OB)=65:M$=«TIENES EL»+W$
1020 RETURN
```

VERBO 12

```
1030 IF RM=43 AND (OB=28 OR OB=29) THEN F(17)=0:M$=«CAJON ABIERTO»
1040 IF RM=28 AND OB=25 THEN M$=«ESTA CERRADO»
1050 IF RM=38 AND OB=32 THEN M$=«¡ESTO ES MISTERIOSO!»:F(2)=0
1060 RETURN
```

VERBO 13

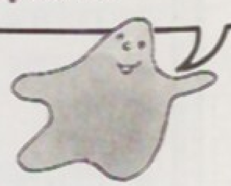
```
1070 IF OB=30 THEN F(18)=0; M$=«¡ALGO AQUI!»
1080 IF OB=31 THEN M$=«¡ES HORROROSO!»
1090 IF (OB=28 OR OB=29) THEN M$=«HAY UN CAJON»
1100 IF OB=33 OR OB=5 THEN GOSUB1140
1110 IF RM=43 AND OB=35 THEN M$=«HAY ALGO DEBAJO...»
1120 IF OB=32 THEN GOSUB 1030
1130 RETURN
```

VERBO 14

```
1140 IF RM=42 AND OB=33 THEN M$=«HAY PALABRAS MAGICAS»
1150 IF (OB=3 OR OB=36) AND C(3)=1 AND F(34)=0 THEN M$=«UTILIZA ESTA PALABRA CON MUCHO CUIDADO "XZANFAR"»

1160 IF C(5)=1 AND OB=5 THEN M$=«EL MANUSCRITO ESTA EN UNA LENGUA EXTRANJERA»
1170 RETURN
```

Utiliza la forma de RND que requiera tu computadora.



VERBO 15

```
1180 M$=«OK»'«+W$'»'
1190 IF C(3)=1 AND OB=34 THEN M$=«PROCESO MAGICO»: IF RM< >45 THEN RM=RND(63)
1200 IF C(3)=1 AND OB=34 AND RM=45 THENF(34)=1
1210 RETURN
```

VERBO 16

```
1220 IF C(12)=1 THEN M$=«HICISTE UN AGUJERO»
1230 IF C(12)=1 AND RM=30 THEN M$=«ARRANCA LOS BARROTES»:D$(RM)=«AGUJERO EN LA PARED»:R$(RM)=«NSE»
1240 RETURN
```

VERBO 17

```
1250 IF C(14) <>1 AND RM=7 THEN M$=«NO HAY TIEMPO PARA JUEGOS»
1260 IF OB=14 AND C(14)=1 THEN M$=«LO LOGRASTE»
1270 IF OB=13 AND C(13)=1 THEN M$=«¡WHOOSH!»
```

```
1280 IF OB=13 AND C(13)=1 AND RMJ=43 THEN R$(RM)=WN:D$(RM)=«ESTUDIO CON PARED SECRETA»:M$=
«ROMPISTE LA PARED»
1290 RETURN
```

VERBO 18

```
1300 IF OB=14 AND C(14)=1 THEN M$=«¡NO ESTA UNIDA A NADA!»
1310 IF OB=14 AND C(14)< >1 AND RM=7 AND F(14)=0 THEN M$=«VES UN BOSQUE DENSO Y UN ACANTILADO AL SUR»:F(14)-1:RETURN
1320 IF OB=14 AND C(14) < >1 AND RM=7 AND F(14)=1 THEN M$=«BAJANDO»: F(14)=0
1330 RETURN
```

VERBO 19

```
1340 IF OB=17 AND C(17)=1 AND C(8)=0 THEN M$=«TE QUEMARA LAS MANOS»
1350 IF OB=17 AND C(17)=1 AND C(9)=0 THEN M$=«NO TIENES NADA CON QUE ENCENDERLA»
1360 IF OB=17 AND C(17)=1 AND C(9)=1 AND C(8)=1 THEN M$=«PRODUCE UNA LUZ VACILANTE»: F(10)=1
1370 RETURN
1380 IF F(0)=1 THEN F(0)=0:M$=«SE APAGO»
1390 RETURN
```

VERBO 21

```
1400 IF OB=26 AND C(16)=1 THEN M$=«HISSS»
1410 IF OB=26 AND C(16)=1 AND F(26)=1 THEN F(26)=0: M$=«¡PFFT! CODIGOS»
1420 RETURN
```

VERBO 22

```
1430 IF OB=10 AND C(10)=1 AND C(11)=1 THEN M$=«ENCENDIDA»:F(24)=1
1440 IF F(27)=1 AND F(24)=1 THEN M$=«¡WHIZZ-ABSORBIDOS LOS FANTASMAS!»: F(27)=0
1450 RETURN
```

VERBO 23

```
1460 IF RM=43 AND (OB=27 OR OB=28) THEN GOSUB1030
1470 IF RM=28 AND OB=25 AND F(25)=0 AND C(18)=1 THEN F(25)=1:R$(RM)=«SEW»:D$(RM)=«GRAN PUERTA ABIERTA»:M$=¡LA LLAVE GIRA!»
1480 RETURN

1490 IF C(OB)=1 THEN C(OB)=0: L(OB)=RM: M$=«HECHO»
1500 RETURN
```

VERBO 25

```
1510 S=0
1520 FOR I=1 TO G
1530    IF C(I)=1 THEN S=S+1
1540    NEXT I
1550 IF S=17 AND C(15)< >1 AND RM < >57 THEN PRINT «TIENES TODO»: PRINT «VUELVE A LA PUERTA PARA VER LA PUNTUACION FINAL»
1560 IF S=17 AND RM=57 THEN PRINT «PUNTUACION DOBLE POR LLEGAR AQUI»: S=S+2
1570 PRINT «TU PUNTUACION=»:S:IF S>18 THEN PRINT «¡ESTUPENDO! TERMINASTE EL JUEGO»: END
1580 INPUT «PRESIONA RETURN PARA CONTINUAR»; Q$
1590 RETURN
```

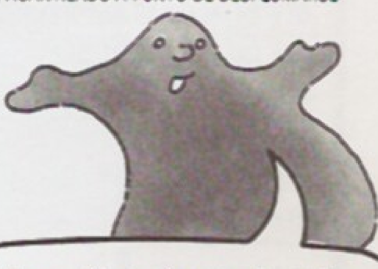


INICIALIZACION

```
1600 DIM R$(63), D$(63), O$(W), V$(V)
1610 DIM C(W), L(6), F(W)
1620 DATA 46, 38, 35, 50, 13, 18, 28, 42, 10,
25, 26, 4, 2, 7, 47, 60, 43, 32
1630 FOR I=1 TO G
1640 READ L(I)
1650 NEXT I
1660 DATA AYUDA, ¿LLEVAR?, IR, N, S, W, E, U, D, RECOGER, TOMAR, ABRIR, EXAMINAR, LEER, DECIR
1670 DATA CAVAR, GOLPEAR, ESCALAR, ENCENDER, APAGAR, FUMIGAR, USAR, DEJAR, PUNTUACION
1680 FOR I=1 TO V
1690    READ V$(I)
1700    NEXT I
```

```
1710 DATASE, WE, WE, SWE, WE, WE, SWE, WS
1720 DATANS, SE, WE, NW, SE, W, NE, NSW
1730 DATANS, NS, SE, WE, NWUD, SE, WSUD, NS
1740 DATAN, NS, NE, WE, WE, NSW, NS, NS
1750 DATAS, NSE, NSW, S, NSUD, N, N, NS
1760 DATANE, NW, NE, W, NSE, WE, W, NS
1770 DATASE, NSW, E, WE, NW, S, SW, NW
1780 DATANE, NWE, WE, WE, WE, NWE, NWE, W
1790 FOR I=0 TO 63
1800     READ R$(I)
1810     NEXT I
1820 DATA ESQUINA OSCURA, JARDIN SIN CUIDAR, MONTON DE LEÑA, PATIO JUNTO A BASURA
1825 DATA PARCELA CON HIERBAJOS, BOSQUE, BOSQUE ESPESO, ARBOL QUEMADO
1840 DATA ESQUINA DE LA CASA, ENTRADA A LA COCINA, COCINA CON UN FOGON VIEJISIMO, PUERTA DE SERVICIO
1845 DATA SALA CON CENTIMETROS DE POLVO, HABITACION DEL TORREON TRASERO, ZONA DESPEJADA JUNTO A LA CASA, CAMINO
1860 DATA LADO DE LA CASA, PARTE TRASERA DEL VESTIBULO, ALCOBA OSCURA, HABITACION OSCURA
1865 DATA ESCALERA DE CARACOL, PASILLO ANCHO, ESCALONES RESBALADIZOS, ALTO DEL ACANTILADO
1880 DATA PARED A PUNTO DE DERRUMBARSE, PASILLO TENEBROSO, ENTRADA DE LUZ, IMPRESIONANTE VESTIBULO ABOVEDADO
1885 DATA VESTIBULO CON UNA GRUESA PUERTA DE MADERA, SALA DE TROFEOS, SOTANO CON UNA VENTANA CON BARROTES, CAMINO HACIA EL
   ACANTILADO
1900 DATA ARMARIO CON ABRIGO COLGADO, VESTIBULO, SALA DE ESPERA, HABITACION SECRETA
1905 DATA ESCALONES ALTOS DE MARMOL, COMEDOR, SOTANO PROFUNDO CON ATAUD, CAMINO HACIA EL ACANTILADO
1920 DATA ROPERO, RECIBIDOR, BIBLIOTECA CON LIBROS MISTERIOSOS, ESTUDIO CON MESA Y UNA AGUJERO EN LA PARED
1925 DATA LUGUBRE HABITACION CON TELARAÑAS, HABITACION MUY FRIA, HABITACION MISTERIOSA, CAMINO DEL ACANTILADO JUNTO A LA MARISMA.
1940 DATA VARANDA CARCOMIDA, PORCHE DELANTERO, TORREON FRONTAL, PASILLO CON PENDIENTE
1945 DATA GALERIA SUPERIOR, MARISMA JUNTO A LA PARED, MARISMA, CAMINO ENCHARCADO
1960 DATA VERJA SEMIDERRUIDA, CAMINO A TRAVES DE UNA PUERTA DE HIERRO, JUNTO A LA VERJA, BAJO LA TORRE
1965 DATA ESCOMBROS PROCEDENTES DEL MURO, MURO CAIDO, ARCO DE PIEDRA EN MALAS CONDICIONES, ACANTILADO A PUNTO DE DESPLOMARSE
1980 FOR I=0 TO 63
1990     READ D$(I)
2000     NEXT I
2010 DATA CUADRO, ANILLO, HECHIZOS MAGICOS, COPA, PERGAMINO, MONEDAS, ESTATUA, CANDELABRO
2012 DATA CERILLAS, ASPIRADORES, PILAS, PALA, HACHA, CUERDA, BARCA, SPRAY, VELA, LLAVE
2014 DATA NORTE, SUR, OESTE, ESTE, ARRIBA, DEBAJO
2016 DATA PUERTA, MURCIELAGOS, FANTASMAS, CAJON, MESA ABRIGO, BASURA
2018 DATA ATAUD, LIBROS, X ZANJAS, PARED, HECHIZOS
2060 FOR I=1 TO W
2070     READ D$(I)
2080     NEXT I
2090 F(18)=1: F(17)=1: F(2)=1: F(26)=1: F(28)=1: F(23)=1: LL=60: RM=57: M$=«OK»
2100 RETURN
```

Observa que los datos están separados por comas. Si cambias los datos asegúrate de no añadir más comas, ya que si no confundirás a la computadora.



Cambiar para el VIC y el Oric

Si tienes un VIC 20 o un Oric usa estos pasos en lugar del paso 460 del listado principal.

```
455 IF VB>14 THEN GOTO 465
460 ON VB GOSUB 500, 570, 640, 640,640, 640, 640, 640, 980, 980, 1030, 1070, 1140
463 GOTO 470
465 ON VB-14 GOSUB 1180, 1220, 1250, 1300, 1340, 1380, 1400, 1430, 1460, 1490, 1510, 1590
```

Cambiar para el Spectrum (Timex 2000)

Los computadores Sinclair (Timex) utilizan una versión de BASIC que difiere bastante del BASIC de otros computadores conocidos; por tanto, tendrás que hacer bastantes cambios para que funcione. Estos cambios hacen que el programa ocupe algo más de 16K. Sin embargo, puedes intentar acortar algunos verbos y mensajes para que se pueda utilizar.

Utiliza los pasos listados a continuación en lugar de los equivalentes del listado principal, también debes realizar los siguientes cambios:

1. El Spectrum necesita LET cada vez que asignas un valor a una variable. Ej.: LET V=25. Esto afecta a muchos pasos, incluyendo aquellos del tipo IF...THEN, por lo que ten mucho cuidado.
2. Todas las cadenas de datos de los pasos 1660, 1665, 1710-1780, 1820-1965 y 2010-2018 deben colocarse entre comillas como: 1820 DATA «ESQUINA OSCURA», «JARDIN SIN CUIDAR», etcétera.
3. En los pasos 1790 y 1980 cambia el bucle para leer FOR I=1 TO 64. (El Spectrum no permite utilizar la etiqueta cero en una matriz.)

```
60 LEFT$=""
120 PRINT D$(RM+1)
140 FOR I=1 TO LEN (R$(RM+1))
150 PRINT R$(RM+1) (I TO I);",";
240 LET X$="": LET W$="": LET VB=0: LET OB=0
250 FOR I=1 TO LEN (Q$)-1
260 IF Q$(I TO I)=" " AND X$="" THEN LET X$=Q$ (TO I-1)
270 IF Q$(I+1 to I+1) <>" " AND X$="" THEN LET W$=Q$(I+1 TO): LET I=LEN (Q$)-1
290 IF W$="" THEN LET X$=Q$
295 IF LEN (X$) > LEN (V$ (1)) OR X$="" THEN GOTO 325
296 LET X$=X$+F$ (TO (LEN (V$(1))-LEN(X$)))
310 IF X$=V$(I) THEN LET VB=I
325 IF WS=" " OR LEN(W$)>LEN (O$(1)) THEN GOTO 360
326 LET W$=W$+F$ (TO(LEN O$(1))-LEN (W$)))
405 IF OB=0 THEN GOTO 420
420 IF (26)-1 AND RM-13 AND INT (RND*3+1)< >3 AND VB < >21 THEN LET M$=«MURCIELAGOS ATACANDO»
430 IF RM=44 AND INT (RND*2)=1 AND F(24)< >1 THEN LET F(27)=1
440 IF F(20)=I THEN LET LL=LL-I
450 IF LL <1 THEN LET F (20)=0
460 GOSUB 500*(VB=1)+570*(VB>2 AND VB>10)+980*(VB=10 OR VB=11)+1030*(VB=12)+1070*(VB=13)+1
140*(VB=14)+1180*(VB=15)+1220*(VB=16)+1250*(VB=17)+1300*(VB=18)+1340*(VB=19)+1380*(VB=20)+
1400*(VB=21)+1430*(VB=22)+1460*(VB=24)+1510*(VB=25)+1590*(VB=26)

800 IF (RM=26 AND F(20)=0) AND (D=1 OR D=4) THEN LET M$=«NECESITAS UNA LUZ»: RETURN
840 IF (RM>26 AND RM>30) AND F(20)=0 THEN LET M$=«DEMASIADO OSCURO PARA MOVERSE»: RETURN
850 LET F(35)=0:LET RL=LEN (R$(RM+1))
870 LET U$=R$ (RM+1) (I TO I)
960 IF RM=41 AND F(23)=1 THEN LET R$(50)=«SW»: LET M$=«LA PUERTA SE CIERRA DANDO UN PORTAZO»: LET F(23)=0
980 IF OB >G OR OB=0 THEN LET M$=«NO PUEDO OBTENER»: W$: RETURN
1190 IF C(3)=1 AND OB=34 THEN LET M$=«PROCESO MAGICO»: IF RM< >45 THEN LET RM=INT (RND*64)
1230 IF C(12)=1 AND RM=30 THEN LET M$=«ARRANCA LOS BARROTES»: LET D$(RM+1)=«AGUJERO EN LA PARED»: LET R$(RM+1)=«NSE»
1280 IF OB=13 AND C(13)=1 AND RM=43 THEN LET R$(RM+1)=«WN»: LET D$(RM+1)=«ESTUDIO CON HABITACION SECRETA»: LET M$=«ROMPISTE LA FINA
PARED»
1360 IF OB=17 AND C(17)=1 AND C(9)=1 AND C(8)=1 THEN LET M$=«PRODUCE UNA LUZ VACILANTE»: LET F(20)=1
1380 IF F(20)=1 THEN LET F(20)=0: LET M$=«SE APAGO»
1470 IF RM=28 AND OB=25 AND F(25)=0 AND C(18)-1 THEN LET F(25)=1: LET R$(RM+1)=«SEW»: LET D$(RM+1)=«GRAN PUERTA ABIERTA»: LET M$=«¡LA
LLAVE GIRA!»
1600 DIM R$(64,4): DIM D$(64,4): DIM O $(W,13): DIM V$(V,9)
```

Recuerda que si introduces los datos en mayúsculas, debes jugar utilizando mayúsculas —la computadora no reconoce que «IR OESTE» e «ir oeste» sean lo mismo—. Es mejor conservar las mayúsculas puestas todo el rato.

Versión para el ZX81 (Timex 1000)

El listado de las seis páginas siguientes es una versión especial de la *Casa Encantada* para el ZX81. Se fija lo más posible a la estructura del listado principal, por lo que puedes seguir las explicaciones del programa que se dan a lo largo del libro. La principal diferencia es que el ZX81 sólo acepta una instrucción en cada paso y no tiene las instrucciones READ...DATA. El programa ha sido escrito de nuevo teniendo en cuenta estas y otras diferencias con el BASIC que se ha tratado en el listado.

Estos cambios absorben gran cantidad de memoria. Por ello para que valga el juego se ha reducido el número de terrenos de 64 a 36 y se han modificado datos. El plano para la versión ZX81 se muestra debajo. En la página siguiente descubrirás cómo se introducen los datos en la memoria de la computadora.

Observa que las casillas se numeran empezando con 1, ya que el ZX81 no te permite utilizar la casilla cero en las matrices.

Plano para el ZX81

1 ENTRADA A LA COCINA	2 COCINA	3 SERVICIO	4 HABITACION POLVORIENTA	HABITACION DEL TORREON TRASERO	6 ROPERO CON ABRIGO
7 PARTE TRASERA DEL VESTIBULO	8 ALCOBA OSCURA	9 HABI-TACION PEQUEÑA CON BASURA	10 ESCALERA DE CARACOL	11 PASILLO ANCHO	12 ESCALONES RESBALADI-ZOS
13 VESTIBULO	14 ENTRADA DE LUZ	15 SALON ABOVEDA-DO	16 SALON CON PUERTA CERRADA	17 SALA DE TROFEOS	18 SOTANO
19 VESTIBULO DELANTERO	20 SALA DE ESPERA	21 HABI-TACION SECRETA	22 ESCALERAS ALTAS DE MARMOL	23 COMEDOR	24 CUARTO CON ATAUD
25 RECIBIDOR DELANTERO	26 BIBLIOTECA	27 ESTUDIO	28 HABI-TACION CON TELAS DE ARAÑA	29 HABI-TACION FRIA	30 SALA TENEBROSA
31 PORCHE DELANERO	32 TORREON DELANTERO	33 PASILLO CON PENDIENTE	34 GALERIA SUPERIOR	35 EMBARCA-DERO	36 CAMINO ENCHARCADO

Cómo usar el programa

Si observas este listado verás que los datos para el juego no están incorporados en el programa. El programa funciona pidiéndote que escribas los datos y luego grabando el programa completo, incluyendo los datos, en cinta. Sólo tienes que hacer esto una vez, la siguiente vez que quieras usar el juego tienes que cargar la cinta.

Sigue estas instrucciones para usar el programa:

1. Copia el programa (con mucho cuidado).
2. Escribe RUN 2440.
3. Ahora escribe los datos en el siguiente orden (ver página 45 para las listas de datos):
   a) descripciones de los terrenos,
   b) salidas,
   c) objetos,
   d) verbos.
   El programa se detiene después de entrar cada sección, por lo que puedes volver a meter cualquier dato que hayas metido mal. Por ejemplo, si quieres volver a introducir los verbos escribe GOTO 2720. Si quieres continuar con la siguiente sección escribe CONT. seguido de NEWLINE.
4. Ahora graba (SAVE) el programa en cinta. Esto graba también todos los datos.
5. Para comenzar el juego escribe GOTO 10. NO ESCRIBAS RUN, ya que esto destruiría todas las variables.
6. Ahora introduce las posiciones iniciales de los objetos. Cuando introduzcas la última de éstas (18), el programa te da tu punto de partida.
7. Para un nuevo juego repite los pasos 5 y 6.
8. Cuando cargues el programa desde la cinta comienza estas instrucciones en el paso 5.

DESCRIPCION Y FEEDBACK

```
10 GOSUB 2200
20 CLS
30 PRINT «AVENTURA DE LA CASA ENCANTADA»
40 PRINT «-------»
50 PRINT «TU POSICION»
60 PRINT D$(RM)
70 PRINT «EXITS»
80 FOR I=1 TO LEN (R$(RM))
90 PRINT R$ (RM) (I TO I), ",";
100 NEXT I
110 PRINT
120 FOR I=1 TO G
130 IF L(I)=RM AND F(I)=0 THEN PRINT «PUEDES VER»; O$(I): «AQUI»
140 NEXT I
150 PRINT «===============»
160 PRINT M$
170 LET M$=«¿QUE?»
```

INPUT Y ANALISIS DEL INPUT

```
180 PRINT «¿QUE HARAS AHORA»
190 INPUT Q$
200 LET X$=""
210 LET W$=""
220 LET VB=0
230 LET OB=0
240 FOR I=1 TO LEN(Q$)-1
250 IF Q$ (I TO I)=" " AND X$="" THEN LET X$=Q$(TO I-1)
260 IF Q$ (I+1 TO I+1)< > " " AND X$< > " " THEN LET W$=Q$(I+1 TO)
270 IF W$< >" " THEN LET I=LEN (Q$)-1
280 NEXT I
290 IF W$="" THEN LET X$=Q$
300 IF LEN(X$>LEN(V$(1)) OR X$="" THEN GOTO 420
310 LET F=LEN (V$(1))-LEN(X$)
320 LET X$=X$+F$(TO F)
330 FOR I=1 TO V
340 IF X$=V$(I) THEN LET VB=I
350 NEXT I
360 IF W$="" OR LEN (W$)>LEN (O$(1)) THEN GOTO 430
```

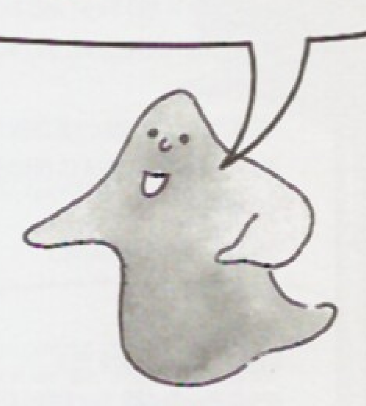
El ZX81 necesita LET cuando se asigna un valor a una variable.

Esta sección parece diferente a la del listado principal debido a que el ZX81 no utiliza MID$, LEFT$ y RIGHT$ para tomar una sección de una cadena.

MENSAJES DE ERROR Y CONDICIONES ESPECIALES

```
370 LET F=LEN (O$(1))-LEN(W$)
380 LET W$=W$+F$(TO F)
390 FOR I=1 TO W
400 IF W$=O$(I) THEN LET OB=I
410 NEXT I
420 IF W$>" " AND OB=0 THEN LET M$=«ESO ES RIDICULO»
430 IF VB=0 THEN LET VB=V+1
440 IF W$=" " THEN LET M$=«NECESITO DOS PALABRAS»
450 IF VB>V AND OB>0 THEN LET M$=«NO PUEDES»+O$
460 IF VB>V AND OB=0 THEN LET M$=«NO TIENE SENTIDO»
470 IF OB=0 OR OB>G THEN GOTO 490
480 IF VB<V AND OB>0 AND C(OB)=0 THEN LET M$=«NO TIENES»+W$
490 IF F(26)=0 OR RM< >5 OR INT (RND*3)=2 OR VB=21 THEN GOTO 520
500 LET M$=«ATACAN LOS MURCIELAGOS»
510 GOTO 20
520 IF RM=28 AND INT (RND*2)=1 AND F(24)=0 THEN LET F(27)=1
530 IF F(20)=1 THEN LET LL=LL-1
540 IF LL<1 THEN LET F (20)-0
```

Observa que algunos de los pasos del programa son más largos que una línea impresa. Ten cuidado de no apretar NEWLINE antes de acabar de copiar el paso.



RAMIFICACION A LAS SUBRUTINAS

```
550 GOSUB 590*(VB=1)+660*(VB=2)+730*(VB)>2 AND VB<10)+1160*(VB=10 OR VB=11)+1270*(VB=12)
+1350*(VB=13)+1440*(VB=14)+1480*(VB=15)+1540*(VB=16)+1560*(VB=17)+1640*(VB=18)+1700*(VB=19)
+1760*(VB=20)+1800*(VB=21)+1850*(VB=22)+1920*(VB=23)+1990*(VB=24)+2040*(VB=25)*(VB=26)
560 IF LL=10 THEN LET M$=«TU VELA SE VA GASTANDO»
570 IF LL=1 THEN LET M$=«SE TERMINO LA VELA»
580 GOTO 20
```

Este paso sustituye el paso ON GOSUB que el ZX81 no posee. Funciona como un cálculo largo, utilizando el valor de VB. La computadora se fija en cada uno de los paréntesis que contiene «VB=» y pone un 1 si el paréntesis es cierto y un cero si no lo es. Intenta ejecutar el cálculo tomando un determinado valor de VB y así verás cómo funciona.

VERBO 1

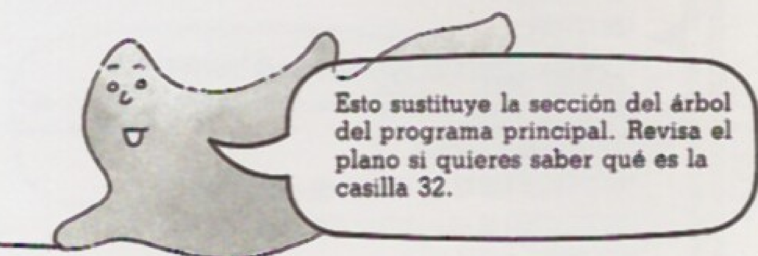
```
590 PRINT «PALABRAS QUE SE»
600 FOR I=1 TO V
610 PRINT V$(I); ",";
620 NEXT I
630 LET M$=" "
640 GOSUB 2160
650 RETURN
```

VERBO 2

```
660 PRINT «LLEVAS:»
670 FOR I=1 TO G
680 IF C(I)=1 THEN PRINT O$(I);",";
690 NEXT I
700 LET M$=" "
710 GOSUB 2160
720 RETURN
```

VERBOS 3-9

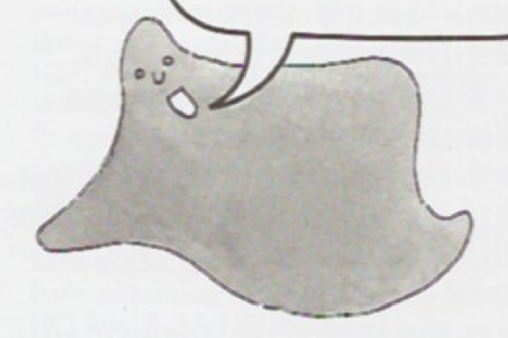
```
730 LET D=0
740 IF OB=0 THEN LET D=VB-3
750 IF OB >18 AND OB <25 THEN LET D=OB-18
760 IF RM=10 AND D=5 THEN LET D=1
770 IF RM=10 AND D=6 THEN LET D=3
780 IF RM=12 AND D=6 THEN LET D=2
790 IF RM=12 AND D=5 THEN LET D=3
800 IF RM=22 AND D=6 THEN LET D=1
810 IF RM=22 AND D=5 THEN LET D=2
820 IF RM<>32 OR D<>3 THEN GOTO 850
830 LET M$=«UNA SERIA CAIDA»
```



```
840 RETURN
850 IF F(27)=0 OR RM<>34 THEN GOTO 880
860 LET M$=«LOS FANTASMAS NO TE DEJAN MOVERTE»
870 RETURN
880 IF RM< >29 OR C(1)=0 OR F(34)=1 THEN GOTO 910
890 LET M$=«UNA BARRERA MAGICA POR EL OESTE»
900 RETURN
910 IF RM<14 OR RM>17 OR F(20)=1 THEN GOTO 950
920 IF RM=14 AND D<>1 AND D<>4 THEN GOTO 950
930 LET M$=«DEMASIADO OSCURO PARA MOVERSE»
940 RETURN
950 IF C(15)=0 OR RM<>36 THEN GOTO 980
960 LET M$=«LA BARCA ES DEMASIADO PESADA»
970 RETURN
980 LET  RL-LEN(R$(RM))
990 LET OM=RM
1000 FOR I=1 TO RL
1010 LET U$=R$ (RM) (I TO I)
1020 IF U$=«N» AND D=1 THEN LET OM=OM-6
1030 IF U$=«S» AND D=2 THEN LET OM=OM+6
1040 IF U$=«W» AND D=3 THEN LET OM=OM-1
1050 IF U$=«E» AND D=4 THEN LET OM=OM+1
1060 NEXT I
1070  M$=«OK»
1080 IF RM=OM THEN LET M$=«NO PUEDES IR POR ESE LADO»
1090 LET RM=OM
1100 IF D<1 THEN LET M$=«IR DONDE»
1110 IF RM<>25 OR F(23)=0 THEN GOTO 1150
1120 LET R$(31)=""
1130 LET M$=«LA PUERTA DA UN PORTAZO DETRAS DE TI»
1140 LET F(23)=0
1150 RETURN
1160 IF OB>0 AND OB<=G THEN GOTO 1190
1170 LET M$=«NO PUEDES RECOGER»+W$
1180 RETURN
1190 IF L(OB< >RM THEN LET M$=«NO ESTA AQUI»
1200 IF F(OB)=1 THEN LET M$=«¿QUE?»+W+«?»
1210 IF C(OB)=1 THEN THEN LET M$=«YA LO TIENES»
1220 IF L(OB)<>RM OR F(OB)=1 THEN GOTO 1260
1230 LET C(OB)=1
1240 LET M$=«TIENES»+W$
1250 LET L(OB)=37
1260 RETURN
1270 IF RM< >27 OR (OB< >28 AND OB< >29) THEN GOTO 1300
1280 LET M$=«CAJON ABIERTO»
1290 LET F(17)=0
1300 IF RM=16 AND OB=25 THEN LET M$=«ESTA CERRADO»
1310 IF RM<>24 OR OB<>32 THEN GOTO 1340
1320 LET M$=«TENEBROSO»
```

VERBOS 10 y 11

VERBO 12







```
1330 LET F(2)=0
1340 RETURN
```

VERBO 13

```
1350 IF OB<>30 THEN GOTO 1380
1360 LET M$=«HAY ALGO AQUI»
1370 LET F(18)=0
1380 IF OB=28 OR OB=29 THEN LET M$=«HAY UN CAJON»
1390 IF OB=33 OR OB=5 THEN GOSUB 1140
1400 IF RM=27 AND OB=35 THEN LET M$=«ALGO DENTRO»
1410 IF OB=32 THEN GOSUB 1270
1420 IF RM=9 AND OB=31 THEN LET M$=«ES HORROROSO»
1430 RETURN
```

VERBO 14

```
1440 IF RM=26 AND OB=33 THEN LET M$=«ES ALGO DEMONIACO»
1450 IF (OB=3 OR OB=36) AND C(3)=1 AND F(34)=0 THEN LET M$=«UTILIZA ESTA PALABRA CON CUIDADO: XZANFAR»
1460 IF C(5)=1 AND OB=5 THEN LET M$=«UNA LENGUA EXTRANJERA»1480 LET M$=«OK»+W$
1470 RETURN
```

VERBO 15

```
1480 LET M$=«OK»+W$
1490 IF C(3)=0 OR OB<>34 THEN GOTO 1520
1500 LET M$=«PROCESO MAGICO»
1510 IF RM<>29 THEN LET RM=INT (RND*36+1)
1520 IF C(3)=1 AND OB=34 AND RM=29 THEN LET F(34)=1
1530 RETURN
```

VERBO 16

```
1540 IF C12=1 THEN LET M$=«FABRICA UN AGUJERO»
1550 RETURN
```

VERBO 17

```
1560 IF C(14)<>1 AND RM=11 THEN LET M$=«NO HAY TIEMPO PARA JUEGOS»
1570 IF OB=14 AND C(14)=1 THEN LET M$=«LO LOGRASTE»
1580 IF OB=13 AND C(13)=1 THEN LET M$=«WHAOSH»
1590 IF OB<>13 OR C(13)=0 OR RM<>27 THEN GOTO 1630
1600 LET R$(RM)=«WN»
1610 LET D$(RM)=«ESTUDIO CON HABITACION SECRETA»
1620 LET M$=«LOGRASTE ATRAVESARLA»
1630 RETURN
```

VERBO 18

```
1640 IF RM< >32 OR C(14) < > 1 OR OB < >14 THEN GOTO 1680
1650 LET M$=«BAJANDO»
1660 LET RM=RM-1
1670 GOTO 1690
1680 LET M$=«¿DONDE IR?»
1690 RETURN
```

VERBO 19

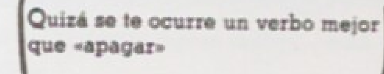
```
1700 IF OB=17 AND C(17)=1 AND C(8)=0 THEN LET M$=«TE QUEMARAS LAS MANOS»
1710 IF OB=17 AND C(17)=1 AND C(9)=0 THEN LET M$=«¿CON QUE?»
1720 IF OB<>17 OR C(17)=0 OR C(9)=0 OR C(8)=0 THEN GOTO 1750
1730 LET M$=«PRODUCE UNA LUZ VACILANTE»
1740 LET F(20)=1
1750 RETURN
```

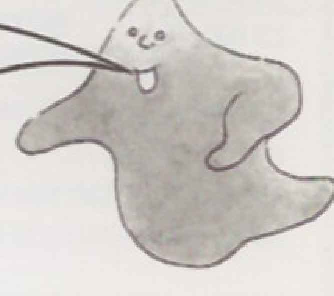
VERBO 20

```
1760 IF F(20)=0 THEN GOTO 1790
1770 LET M$=«SE APAGO»
1780 LET F(20)=0
1790 RETURN
1800 IF OB=26 AND C(16)=1 THEN LET M$=«HISSS»
```









Los pasos 2300 a 2350 asignan las banderas para los objetos invisibles, por lo que no tienes que introducir los datos de las banderas por separado.



Cuando DIMensiones una matriz en el ZX81, debes indicar a la computadora la longitud de la palabra más larga que vayas a almacenar. Entonces la computadora reserva 36 (o los que sean) espacios de esta longitud. El tener una palabra mucho más larga que todas las demás derrocha memoria.

```
1810 IF OB<>26 OR C(16)<>1 OR F(26)=0 THEN GOTO 1840
1820 LET M$=«PFFT - COGIDOS»
1830 LET F(26)=0
1840 RETURN
```

VERBO 22

```
1850 IF OB<>10 OR C(10)=0 OR C(11)=0 THEN GOTO 1910
1860 LET F(24)=1
1870 LET M$=«ENCENDIDA»
1880 IF F(27)=0 OR F(24)=0 THEN GOTO 1910
1890 LET M$=«LAS ABSORBISTE»
1900 LET F(27)=0
1910 RETURN
```

VERBO 23

```
1920 IF RM=27 AND (OB=27 OR OB=28) THEN GOSUB 1270
1930 IF RM<>16 OR OB<>25 OR F(25)=1 OR C(18)=0 THEN GOTO 1980
1940 LET F(25)=1
1950 LET M$=«LA LLAVE GIRA - CLUNK»
1960 LET R$(RM)=«SEW»
1970 LET D$ (RM)=«GRAN PUERTA ABIERTA»
1980 RETURN
```

VERBO 24

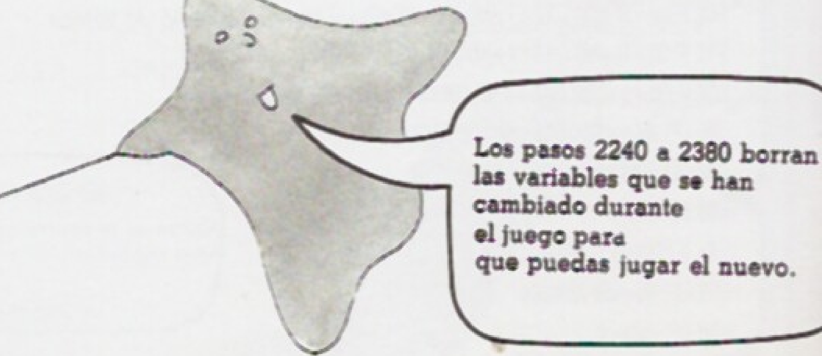
```
1990 IF C(OB)=0 THEN GOTO 2030
2000 LET C(OB)=0
2010 LET M$=«HECHO»
2020 LET L(OB)=RM
2030 RETURN
```

VERBO 25

```
2040 LET S=0
2050 FOR I=1 TO G
2060 IF C(I)=1 THEN LET S=S+1
2070 NEXT I
2080 IF S=17 AND C(15)=0 AND RM<>31 THEN PRINT «TIENES TODO, VUELVE AL PORCHE PARA VER LA PUNTUACION FINAL»
2090 IF S<>17 OR RM<>31 THEN GOTO 2120
2100 PRINT «PUNTUACION DOBLE»
2110 LET S=S+2
2120 PRINT «TU PUNTUACION»
2130 IF S<18 THEN GOTO 2160
2140 PRINT «BIEN HECHO - HAS TERMINADO»
2150 STOP
2160 PRINT «APRIETA NEWLINE PARA CONTINUAR»
2170 INPUT Q$
2180 LET M$=«OK»
2190 RETURN
```

INICIALIZACION

```
2200 FOR I=1 TO W
2210 LET F(I)=0
2220 LET C(I)=0
2230 NEXT I
2240 LET R$(31)=«N»
2250 LET R$(27)=«W»
2260 LET R$(16)=«WE»
2270 LET D$(27)=«ESTUDIO, MESA Y PARED»
2280 LET D$(16)=«VESTIBULO CON UNA GRAN PUERTA DE MADERA»
```



SECCION DE ENTRADA DE DATOS

```
2290 LET M$=«OK»
2300 LET F(18)=1
2310 LET F(17)=1
2320 LET F(27)=1
2330 LET F(2)=1
2340 LET F(26)=1
2350 LET F(23)=1
2360 LET LL=60     13 ESPACIOS
2370 LET RM=31
2380 LET F$=""
2390 FOR I=1 TO G
2400 PRINT I
2410 INPUT L(I)
2420 NEXT I
2430 RETURN
2440 DIM R$(36,4)
2450 DIM D$(36,30)
2460 LET V=25
2470 DIM V$(V,9)
2480 LET W=36
2490 DIM O$(W,13)
2500 DIM C(W)
2510 DIM F(W)
2520 LET G=18
2530 DIM L(G)
2540 PRINT «DESCRIPCIONES»
2550 FOR I=1 TO 36
2560 PRINT I
2570 INPUT D$(I)
2580 NEXT I
2590 STOP
2600 PRINT «SALIDAS»
2610 FOR I=1 TO 36
2620 PRINT I
2630 INPUT R$(I)
2640 NEXT I
2650 STOP
2660 PRINT «OBJETOS»
2670 FOR I=1 TO W
2680 PRINT I
2690 INPUT O$(I)
2700 NEXT I
2710 STOP
2720 PRINT «VERBOS»
2730 FOR I=1 TO V
2740 PRINT I
2750 INPUT V$(I)
2760 NEXT I
2770 STOP
```

Los datos: Introduce los datos en este orden. (Ver página 40.)

DESCRIPCIONES

ENTRADA A LA COCINA, COCINA CON FOGON ANTIGUO, SERVICIO, HABITACION POLVORIENTA, HABITACION DEL TORREON TRASERO, ROPERO CON ABRIGO, PARTE TRASERA DEL VESTIBULO, ALCOBA OSCURA, HABITACION PEQUEÑA CON BASURA, ESCALERA DE CARACOL, PASILLO ANCHO, ESCALONES RESBALADIZOS, VESTIBULO, ENTRADA DE LUZ, VESTIBULO ABOVEDADO, VESTIBULO CON UNA GRAN PUERTA DE MADERA, SALA DE TROFEOS, SOTANO, VESTIBULO DELANTERO, SALA DE ESPERA, HABITACION SECRETA, ESCALERAS ALTAS DE MARMOL, COMEDOR, CUARTO CON ATAUD, RECIBIDOR DELANTERO, BIBLIOTECA CON LIBROS MACABROS, ESTUDIO CON MESA, AGUJERO EN LA PARED, HABITACION CON TELAS DE ARAÑA, HABITACION FRIA, SALA TENEBROSA, PORCHE DELANTERO, PARTE ALTA DEL TORREON DELANTERO, PASILLO CON PENDIENTES, GALERIA SUPERIOR, EMBARCADEROS, CAMINO ENCHARCADO.

SALIDAS

SE, OE, O, SE, OE, O,
NS, SE, OE, NOUD, SE, SOUD,
NS, NSE, OE, OE, NSO, NS
NSE, NSO, S, NSUD, N, N
N, NE, O, NSE, OE, SO
N, OE, OE, NN, E, NO

Las comas indican cuándo debes apretar NEWLINE detrás de cada término. No escribas las comas.

OBJETIVOS

CUADRO, ANILLO, HECHIZOS MAGICOS, COPA, PERGAMINO, MONEDAS, ESTATUA, CANDELABRO, CERILLAS, ASPIRADORA, PILAS, PALO, HACHA, CUERDA, BARCA, SPRAY, VELA, LLAVE, NORTE, SUR, OESTE, ESTE, ARRIBA, ABAJO, PUERTA, MURCIELAGOS, FANTASMAS, CAJON, MESA, ABRIGO, BASURA, ATAUD, XZANFAR, PARED, HECHIZOS

VERBOS

AYUDA, LLEVAR, IR, N, S, O, E, U, D
RECOGER, TOMAR, ABRIR, EXAMINAR, LEER, DECIR, CAVAR, GOLPEAR, ESCALA., ENCENDER, APAGAR, FUMIGAR, USAR, ABRIR, DEJAR, PUNTUACION

SITUACION INICIAL DE LOS OBJETOS

(Debes introducir estos datos cada vez que ejecutas el programa.)

30, 24, 21, 32, 5, 8
16, 26, 2, 13, 14, 36,
18, 11, 35, 3, 27, 6

Trucos y consejos

1. Utiliza variables enteras

En algunas computadoras puedes poner un signo % detrás de una variable numérica para indicar que sólo quieres introducir números enteros (es decir, sin nada detrás de la coma de decimales). La variable V se convierte en V% e igual sucede con las demás. Puedes hacer esto en el BBC, el TRS-80, el Dragón, el TRS-Color y el Oric. Esto es útil, ya que ahorra memoria aumenta la velocidad hasta un 50%. Esta velocidad se observa sobre todo cuando la computadora ejecuta largos bucles.

2. Presentación de la pantalla

Si tienes suficiente memoria disponible puedes intentar mejorar la forma en que la aventura aparece en la pantalla. Puedes añadir gráficos para el comienzo y hacer que el texto se ilumine intermitentemente en algunos momentos del juego, como cuando la vela se apaga o cuando aparecen los fantasmas. El texto no tiene por qué aparecer al borde de la pantalla; ni ha de ser todo del mismo color. Puedes utilizar bordes de colores o bien fondos diversos.

3. Atento a la ortografía

Si no estás muy seguro sobre cómo se escribe una palabra que quieras introducir en el juego, compruébalo en un diccionario. Tu computadora no sabe ortografía y almacena todo lo que tú le escribes. Esto puede ser desmoralizador para el jugador que introduce una instrucción correcta y la computadora se la rechaza por no reconocer la palabra.

4. Dispara la acción

Algunas aventuras resultan aburridas porque todo sucede en el mismo sitio. Procura que haya cosas interesantes por todo el juego.

5. Utiliza instrucciones REM

Si escribes un programa largo y complicado como una aventura, suele ser conveniente introducir instrucciones REM delante de cada sección. Si no lo haces es muy probable que te confundas cuando quieras saber cuál es cuál. Sin embargo, cuando hayas terminado el programa borra las REM, ya que estás ocupando memoria.

6. Usa nombres de variables con sentido

Da nombres a las variables para que luego resulte fácil deducir a qué se refieren. Por ejemplo, OB para los objetos, etcétera. Si tienes suficiente memoria disponible y tu computadora te lo permite es aconsejable utilizar nombres de variables largos para que te sea más fácil recordar a qué se refieren. Por ejemplo, en lugar de V usar verbo. No obstante, haz una lista de tus variables y de lo que significan, para que no las mezcles mientras escribes el programa.

7. Hazlo sencillo

No seas demasiado ambicioso en tus primeros juegos. Un juego simple y bien pensado es más divertido que un juego confuso y complicado. No todo el mundo quiere jugar a un juego que tarda días.

8. Hazlo amistoso

Cuando hayas escrito el juego revisa los comentarios para asegurarte de que no son ambiguos o confusos. En lugar de «DEMASIADO OSCURO» podría decir, por ejemplo: «NECESITAS UNA LUZ PARA ENTRAR AQUI». Recuerda que algo que a ti te resulte evidente puede no serlo para otros jugadores. Haz algunos de tus comentarios divertidos, ya que esto hace que el jugador piense que la computadora está realmente hablando con él.

Respuesta a los problemas

Problema detectivesco (página 13)

Estas son algunas sugerencias para el problema detectivesco. Contrástalas con las que tu pensaste.

1. Sólo se verá el pelo sin ordenar a la computadora que examine el abrigo. No puedes llevarte el pelo a menos que tengas un sobre limpio donde meterlo

2. Necesitas una llave para abrir el cajón, una lupa para ver el hilo y un segundo sobre para meterlo.

3. Necesitas escayola y un recipiente con agua para hacer el molde de la huella.

4. Necesitas polvos de talco para resaltar las huellas y cinta adhesiva para grabarlas desde cualquier superficie.

5. Necesitas un equipo portátil de análisis de sangre (descrito en el juego como una caja de botellas y equipo científico).

Rompecabezas en la aventura (página 15)

Recuerda que no hay respuestas «correctas» a estos problemas. Estas son algunas soluciones sugeridas.

1. Levanta la alfombra y encuentra una trampilla.

2. Usa el pañuelo como máscara (suponiendo que la somnolencia la produce algún gas), busca en el macuto y encuentra un frasco. Abre el frasco y encuentra café negro. Bebe café.

3. Lee el pergamino (es una proclamación de la libertad para la esclava).

4. Tira el postre (da la casualidad de que es un pastel) a la cara del villano. Toma el control remoto y escapa.

Problema (página 28)

Esta es una forma de cambiar el programa para limitar el número de objetos que se pueden llevar a un tiempo.

Necesitas dos nuevas variables que llamaremos OL (Objetos llevados) y LL (Límite para llevar). Añade esto al final del paso 70, de esta forma 70 OL=0: LL=8.

A continuación debes indicarle a la computadora que sume uno a OL en la rutina RECOGER, cuando el jugador tome un objeto, y le reste uno al desarrollarse la rutina DEJAR, siempre que deje un objeto. Haz esto añadiendo respectivamente a los pasos 1010 y 1490:

1010...: OL=OL+1
1490...: OL=OL—1

Ahora añade un nuevo paso a la rutina RECOGER para que antes de continuar con el resto de la rutina compruebe si OL es igual al límite. (LL no tiene por qué ser 8, pero no debe ser inferior, ya que si no el jugador no podría llevar todos los tesoros al finalizar.)

Indice

Impreso en España - Printed in Spain
Melsa, Pinto (Madrid)
Depósito legal: M–32348–1985
I.S.B.N. 84–7374–137–4

Colección Electrónica













